AN

INTERNSHIP REPORT

ON

A RESPONSIVE WEB VIEW APPLICATION FOR ORPHANGE CONNECTING AND ADOPTION

BY

KAMAL ACHARYA

(Tribhuvan University)

Date: 2025/01/25

“A responsive web view application for orphanage connecting and adoption”

Abstract

Orphanage Connecting System is being developed for the welfare of the orphanage and can be used by NGOs for co-operative social work. System is mainly for the systematic usage of the website by the user, adopters, charity or NGO’s and the admin. The objective of this application is to develop a centralized website for orphanages. To provide facilities of various social activities in a single website. People can adopt the children and they can donate through online.

Introduction

As Orphanages are increasing in day to basis simultaneously the helping hands are also increasing in higher range. The practical difficulty is finding out the real orphans and the less knowledge about the donation details. So the website was developed to help to reach the needy orphans by the volunteering donations. It allows the users to search and find the orphanages on internet and then they can donate with the orphanages whichever they want.Orphanage Management System Based Web App. The online orphanage connecting system is developing to facilitate the orphanages. Here the user’s visit the website by registering their details. After that the registration process got over they can enter into the site. They can view the orphanage details .those who is willing to donate to the orphanage can donate amount by giving their details such as name, account number etc. As soon as admin receives that particular amount from the user, response message will be sent to the user’s mail. User can also adopt the orphans by giving their details and send the request for charity and NGO’s. Amount which is given by the users will be sent to the admin account. Admin will transfer that particular amount to the orphanages which have been registered. Adding the children to the orphanages, viewing the user details, updating the data base these are the jobs of the admin. Adoption option is also included with the eligibility checker. This is for those who wish to adopt the child from the orphanage.

Objectives

The main objective of charity management system is that it mainly focuses on helping needy people by collecting extra healthy food from different parties and delivers it to needy people after packing. It also helps in holding campaign and each and every update of money spend in different aspects are send to the donors. The volunteers registration and sending message when the campaign is going to held any places. Their process of donation is that the donator should contact the charity and tell them about the type of the food and best suitable time and date to collect the food.

Existing system

Existing Systems consists of only the orphanages details such as, address contact numbers etc.. It does not have any centralized and separate website for orphanages and also there is difficult find orphan child to adoption. In existing system mainly popular NGO's can get benefit sometimes people can difficult to know about charity information and NGO's. If people adopt orphan child there is no monitoring information about child from adopted users to charities

Drawbacks In existing system

People can view only the address of the particular orphanages which are searched by the users. In that people cannot do any donation or sponsorships. This is the main drawback of the existing system. And also this is the main disadvantage of the orphanages those who are in need of getting help from others.

Proposed system

The proposed system is implemented for the orphanages in Coimbatore premises. It can be extended to state level or national level. Other organizations such as old age home, relief camps also can be included in this project. The details of each child such as medical, education can be added to inform their sponsors. The admin should remotely use the web portal or the software to handle the request of different users and organization. User can also find orphan and adopt them, one who adopt the child they need to update the instant health and educational information. In this application user can also send

Modules

- Login / registration
- Admin
- Charity or NGO
- Public donators / adopters
- Child monitoring
- Nearby donation seekers

Hardware Requirements

RAM	:	2 GB
Hard disk	:	100 GB
Process	:	32/64 bit Pentium

Software Requirements

IDE	:	WAMP
Database	:	MySQL
Technology	:	HTML CSS BOOTSTRAP PHP.

Introduction

1.1 Project overview

The sharp increase in the amount of wastage in terms of food makes the need for charity in terms of donation. In the current scenario food is being wasted daily on a large basis in different restaurants, weddings, social functions, college canteens and many other social events. This projects also helps in helping campaigns like blood donation or creating awareness among the food. People donate food manually by visiting each organization number of times in order to reduce the problems of food wastage where as there are websites that have taken efforts to help people donate food .The proposed system presents, a new internet-based application that provides a platform for donating leftover food to all needy pe ople/organizations. The system is shown to be an effective means of donating things to organizations, etc. over the internet. It shows the potential for avoiding the wastage of food. It provides information about how the product works for betterment of the society. This system will create a common collaboration portal for hotels/restaurants and charities, charity can directly contact restaurants who have food remaining. This product is a web application which aims to establish a link between restaurants and the charity homes/needy households to enable excess food donation.

System Study

2.1 Existing System

In existing system some people or organizations like courtiers, hotels and other NGO's are difficult to find proper donators and they can't seek proper way to reach the needy people. In India there are 10 lack peoples suffering from without food and around 1lack tons of food wasting daily because no proper plan to handle excess of food.

2.1.1 Disadvantages of existing system:

- People don't have plan for excess of food management
- No proper communication between donors and needier
- Manual view contact information of charities or NGO's

2.2 Proposed System:

In the proposed system the donator registers them self in the system and fill up the necessary information along with the contact number and donation things type, the admin can contact donors through the system or by the phone in order to achieve the required donation. Overall, the proposed system is useful for the admin and charities to know the contact number of the donators.

2.2.1 Advantages of proposed system:

- Donators can easily register to application
- Update donation things like food, cloth, etc.
- Each and every update to the donor about the spend money.
- Search nearby food freezers to dump excess of food
- Charities or NGO's can easily request for needed things to admin
- Easily to communication between donators and needier
- If any function is organized in charity, it sends notification to all donors about the
- function details like where the fun is organized, at which time, date etc..

Literature Survey

In this application mainly propses donatoion activity. Any NGOs or one who need donation they can request in this appliaction. By using this application user also donate the things which they have or they can also donate things for caause purpuse. This paper presents ‘Helping Hands’, a new internet-based application that provides a platform for donating old stuff and leftover food to all needy people/organizations. It provides information about the motivation to come up with such an application, thereby describing the existing donation system and how the proposed product works for the betterment of society. The product is shown to be an effective means of donating things to organizations, etc. over the internet. It shows the potential for avoiding the wastage of food, clothes, books and other stuff.

In this application they have developed a mobile application to assist users in managing their food inventory. The application would store and display basic information about the inventory contents and alerts the user of the food products which are due to expire the next day. Consequentially, users may take actions to avoid the concerned products get wasted or spoiled. It is believed that a considerable amount of food waste would be avoided in households if the occupants are well-informed of the timeline of their food stocks. Provisions have also been made to allow for the multi-device use.

Software requirement and specifications

4.1 Purpose

The main purpose of this application include reduction in wastage of food, making food , clothes etc. available to orphanages, old age homes and other such organizations, which will also inculcate values of sharing and sensitivity among people.

4.2 Scope

A Software Requirements Specification (SRS) is a description of a software system to be developed. It lays out functional and non-functional requirements, and may include a set of use cases that describe user interactions that the software must provide.

The software requirements specification document enlists enough and necessary requirements that are required for the project development. To derive the requirements, the developer needs to have clear and thorough understanding of the products to be developed or being developed. This is achieved and refined with detailed and continuous communications with the project team and customer till the completion of the software.

The introduction of the Software Requirements Specification (SRS) provides an overview of the entire SRS with purpose, scope, definitions, acronyms, abbreviations, references and overview of the SRS. The aim of this document is to gather and analyse and give an in-depth insight of the complete A Responsive Application for Client and user by defining the problem statement in detail.

4.3 System analysis

The sharp increase in the amount of wastage in terms of food makes the need for charity in terms of donation. In the current scenario food is being wasted daily on a large basis in different restaurants, weddings, social functions, college canteens and many other social events. This projects also helps in helping campaigns like blood donation or creating awareness among the food. People donate food manually by visiting each organization number of times in order to reduce the problems of food wastage where as there are websites that have taken efforts to help people donate food .The proposed system presents, a new internet-based application that provides a platform for donating leftover food to all needy pe ople/organizations. The system is shown to be an effective means of donating things to organizations, etc. over the internet. It shows the potential for avoiding the wastage of food. It provides information about how the product works for betterment of the society. This system will create a common collaboration portal for hotels/restaurants and charities, charity can directly contact restaurants who have food remaining. This product is a web application which aims to establish a link between restaurants and the charity homes/needy households to enable excess food donation.

4.3.1 Drawbacks of existing system

- People don't have plan for excess of food management
- No proper communication between donors and needier
- Manual view contact information of charities or NGO's

4.3.2 Analysis of proposed system

- Donators can easily register to application
- Update donation things like food, cloth, etc.
- Each and every update to the donor about the spend money.
- Search nearby food freezers to dump excess of food
- Charities or NGO's can easily request for needed things to admin
- Easily to communication between donators and needier
- If any function is organized in charity, it sends notification to all donors about the
- function details like where the fun is organized, at which time, date etc..

4.3.3 Feasibility Study:

A *feasibility study* is an *analysis* used in measuring the ability and likelihood to complete a project covid plasma donation application successfully including all relevant factors.

Depending on the result of the initial investigation, the survey is expended to a more detailed feasibility study. A feasibility study is a test of system proposal according to its work ability impact on the orphanage management application and effective use of resources

Operational Feasibility:

In the orphanage management application there are multiple operations are to be performed to so the application to be more effective and more reliable. Since the application developed by using responsive web technology user can easily operate using rich user interface. User can search nearby donator by tracking current location and donate. User always wants his system to be more effective and more responsive, so considering all the above scenarios to develop system with more effective we choose best tools for projects.

Economic Feasibility:

The main aim of this orphanage management application is to cost effective. The cost to develop this application is less. Because web application can operate using browser. user no need to find orphanage by visiting physically this required more cost and travel expensive. This application enables less cost for other operation.

Motivational feasibility:

The motivational feasibility provides the effective user interface and helps the developer to motivate. Every people have mobile phone hence user can easily operate this application easily from any where from the globe. User can also get instant information about orphanage management application information.

Schedule feasibility

This application is web-based application using web technology it gives tremendous output with less time, i.e., this covid plasma donation application is compatible for all devices with rich user interface hence it helps for easy to operate and took less time.

4.4.1 System goals

- Easy to donate excessive food to needy people
- Each and every update to donor about the money spend
- Viewing the every student details at both side of donor and admin. If the donor wants to donate education or any related information then they can.
- Transparent communication channel between donors and needier
- Search nearby warehouse and freezers to dumb materials and foods
- Get motivate and help others

4.5 General requirements

To be used efficiently, all computer software needs certain hardware components or the other software resources to be present on a computer. These pre-requisites are known as (computer) system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended. With increasing demand for higher processing power and resources in newer versions of software, system requirements tend to increase over time

4.6 Functional Requirements:

1 Donor

- Login / registration

All user roles need to register their details and login to perform operation

- Update donation details

Public or any organization like function hall, showroom, and hotels are update a donation items it may be a cloths or food etc.

- Search nearby freezer / warehouse

People can easily search nearby freezer and warehouse to donate food and other materials

- View the all the information of spend money in charity
- Accepting bill receipt from charity

2 Users

- Send request to admin

Charities and NGO's can easily send required materials request to admin

- Update requirements

Charities can update their requirements to website

3 Admin

- View donation details

Admin can view the donation details from donators and get into warehouse

- Check stocks in warehouse

Admin can check material or food stocks in warehouse

- Accept users request

Admin can view the user request and accept the request and supply the requirements needed to the user

- Provide bill receipt to the Donor:
  The donor who wants to donate for the charity after donating they are provided with bill receipt. To shoe the income tax department
- Each and every details of Students

The students in the charity have each and every details like the place, school, birth.

- Campaign

The Charity can go for Campaign like blood donation or awareness of food in districts. So such each and every update provided to the Donor

- Volunteers

Registration of volunteer based on places. The volunteers will get notification when the charity wants to conduct any campaign

- Provide receipt to donors

Admin can provide proper receipt to donors who give money as donation to charity to avoid the income tax to donors

- Provide detailed information

Charity can provide detailed information to donors about their money who gives as donation to charity to avoid the misuse of money

- Certificate to donors

Charity provide certificate to donors for their service

4 Volunteers

- Login / registration

Admin request the volunteers, if they are interested they can registered otherwise not

4.7 Non Functional Requirement:

- **Availability**

  The degree to which a system, subsystem or equipment is in a specified operable and committable state at the start of a mission, when the mission is called for at an unknown, *i.e.* a random, time. Simply put, availability is the proportion of time a system is in a functioning condition. This is often described as a mission capable rate. Mathematically, this is expressed as 100% minus unavailability.

  This is a Web application and will be available for the user whenever they want to use.

- **Maintainability**

  Maintainability is a characteristic of design and installation, expressed as the probability that an item will be retained in or restored to a specified condition within a given period of time, when the maintenance is performed in accordance with prescribed procedures and resources.

  The project is developed using an Open source tool and is easy to maintain.

- **Security**

  Security is one or more requirements about protection of your system and its data. The measurement can be expressed in a variety of ways like effort, skill level, time etc. to break into the system.

  This feature is provided by having user authentication i.e., providing the user with login-id and password.

- **Reliability**

  Reliability is a requirement about how often the software fails. The measurement is often expressed in mean time between failures. The definition of a failure must be clear. Also, don't confuse reliability with availability which is quite a different kind of requirement. Be sure to specify the consequences of software failure, how to protect from failure, a strategy for error detection, and a strategy for correction. The project maintains data integrity. Database crashes and misuse may affect a user's history.

- **Simplicity**

  The project is driven by a simple user interface which helps to interact easily with application and easy to remember. Application builds by using bootstrap technology hence it compatible for all devices.

4.8 External interface requirement

4.9 Hardware Requirements

- RAM : 2 GB
- Hard disk: 100 GB
- Processor: 32/64 Pentium

4.10 Software Requirements

• Operating system	:	Windows 7
• Front End	:	HTML, CSS, Bootstrap.
• Coding language	:	PHP.
• Software	:	WAMP Server.
• IDE	:	Notepad++.
• Data base	:	MYSQL.

SYSTEM DESIGN

Introduction

The Software Design Document is a document to provide documentation which will be used to aid in software development by providing the details for how the software should be built. Within the Software Design Document are narrative and graphical documentation of the software design for the project including use case models, sequence diagrams, collaboration models, object behaviour models, and other supporting requirement information.

Purpose

The purpose of Software Design Specification (SDS) document is to specify high level view of the architecture of our system, and on the interaction between the user and the system. And another purpose is on detailing a low-level view of each component of the software and how the components interact with each other.

This document's purpose is to provide a high-level design framework around which to build our project A responsive application for deaf and dumb. It also provides a list of requirements against which to test the final project and determine whether we were able to successfully implement the system according to design.

Scope

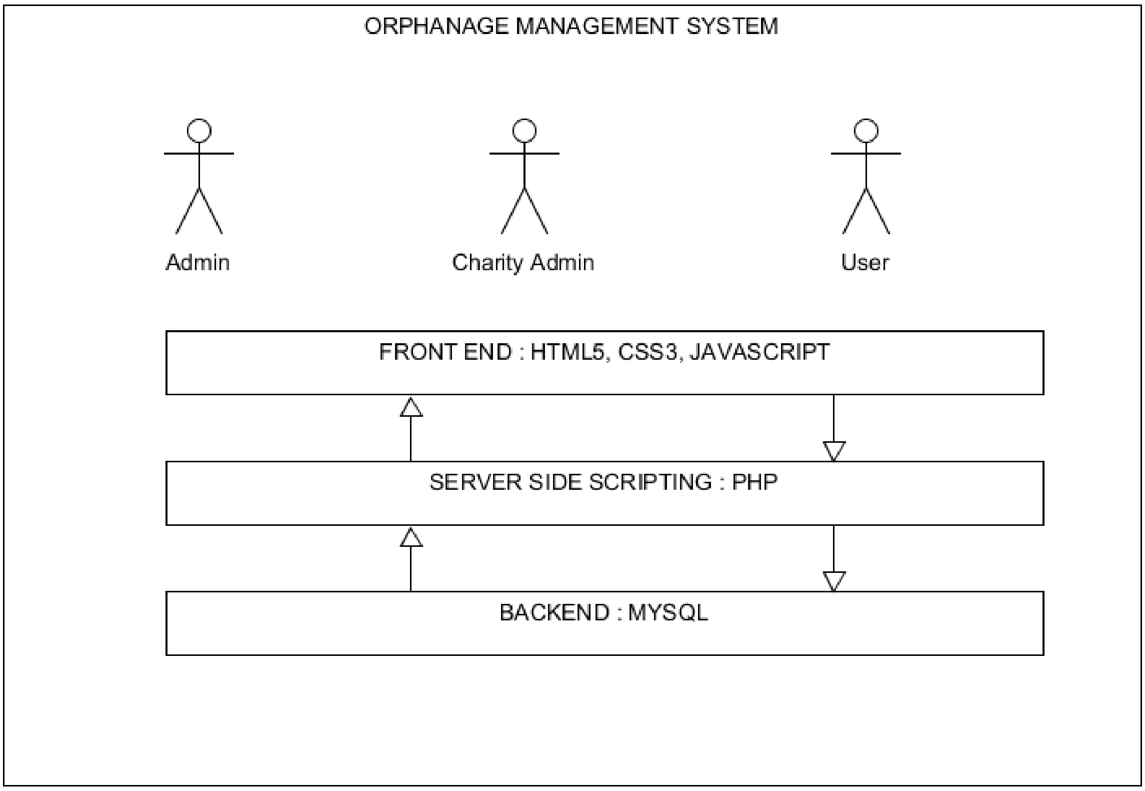
The system Design (SD) describes how the functional and non-functional requirements gathered in the requirement phase, preliminary user-oriented functional design transform into more technical system specifications from which the system is built. This phase describes the design goals and considerations, provides a high-level overview of the system architecture, and describes the data design associated with the human-machine interface and operational scenarios.

Overview

The system design documents and tracks the necessary information required to effectively define architecture and design of orphanage Application system in order to give the development team guidance on the architecture of the system to be developed. Design documents are incrementally and iteratively produced during the system development life cycle, based on the particular circumstances of the Request Approval Application project.

System Architecture

Architecture focuses on looking at a system as a combination of many different components, and how they interact with each other to produce the desired result. It involves the process of defining a collection of hardware and software components and their interfaces to establish the framework for the development of an android application.

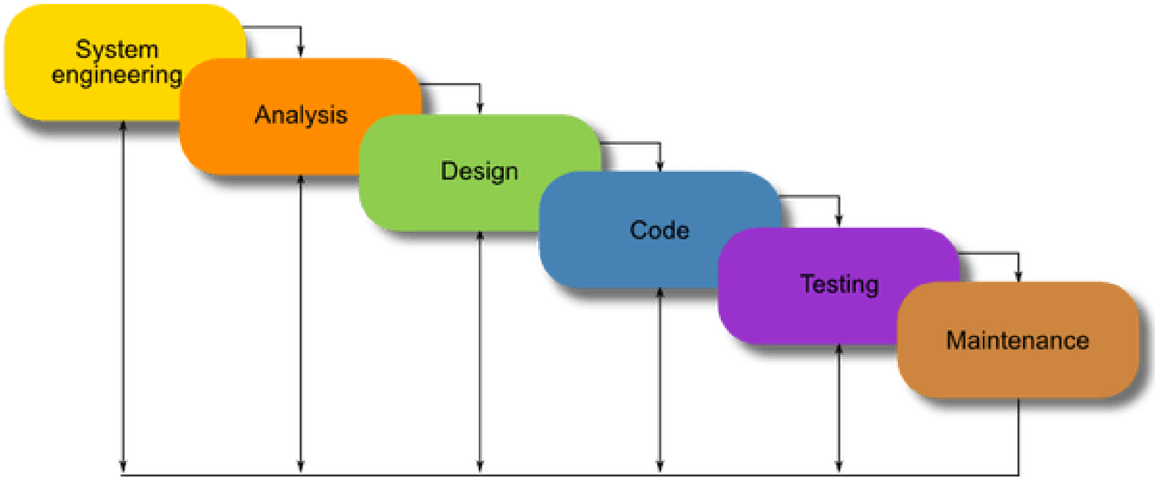

ORPHANAGE MANAGEMENT SYSTEM
Admin
Charity Admin
User
FRONT END : HTML5, CSS3, JAVASCRIPT
SERVER SIDE SCRIPTING : PHP
BACKEND : MYSQL

Architecture Diagram

Waterfall Model

Waterfall model is the earliest SDLC approach that was used for software development. It is also referred to as a linear-sequential life cycle model. It is very simple to understood and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in phases.

Following is a diagrammatic representation of distinct phases of waterfall model.



Waterfall Model

In “The Waterfall” approach, the full process of software development is divided into separate phases. In Waterfall model, typically, the outcomes of one phase act as the input for the next phase sequentially. The sequential phases in Waterfall model are:

- **Requirement Gathering and analysis**

All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document. Considering this project what are all tools and technoly finalised to develop responsive web application and also we finalised who are all the actors belongs to this application.

System design

Considering this project what are all tools and technoly finalised to develop responsive web application and also we finalised who are all the actors belongs to this application.

The requirement specifications from first phase are studied in this phase and system design is prepared. Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.

- **Implementation**

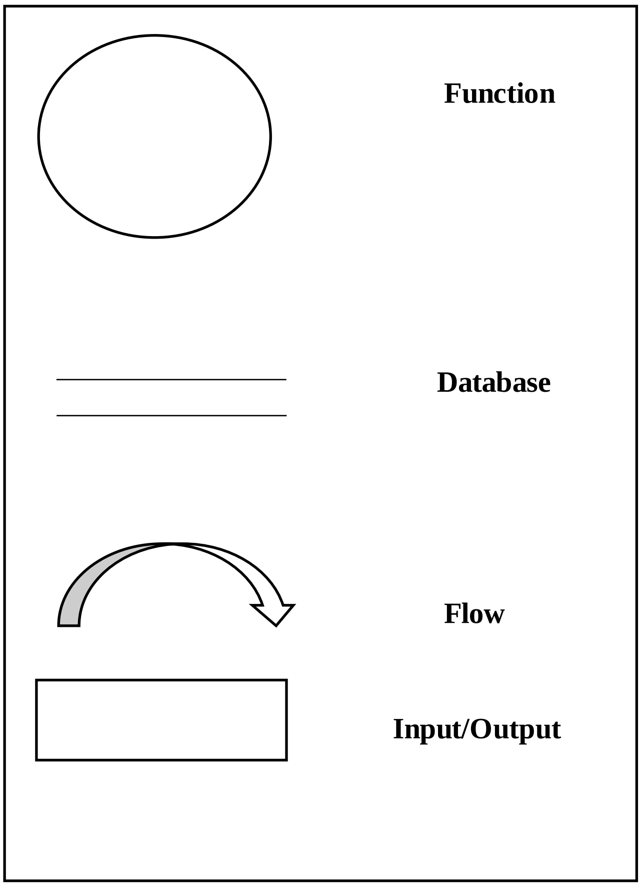
Considering implementation front end design UI created by using HTML, Bootstrap and CSS. For server side script we used PHP and to store data we used Mysql server. By iseing XAMPP tool we implement this applocation.

- **Integration and Testing**

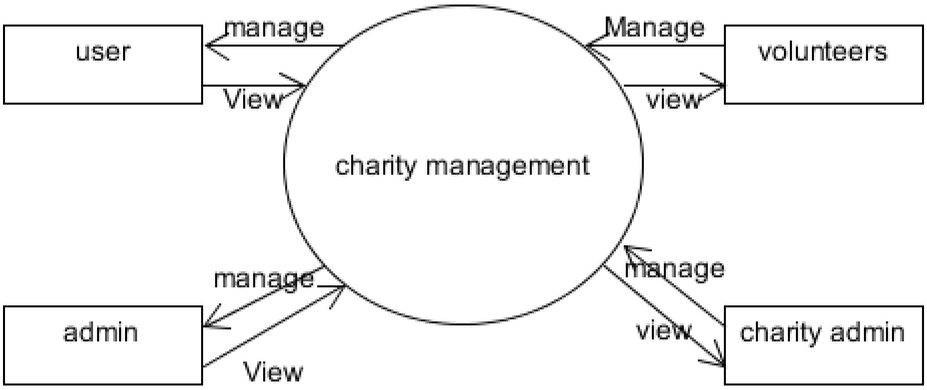
Once application is done condering allactivity we going to apply the software testing activity like unit test, integration test and other test casees.

Data Flow Diagram

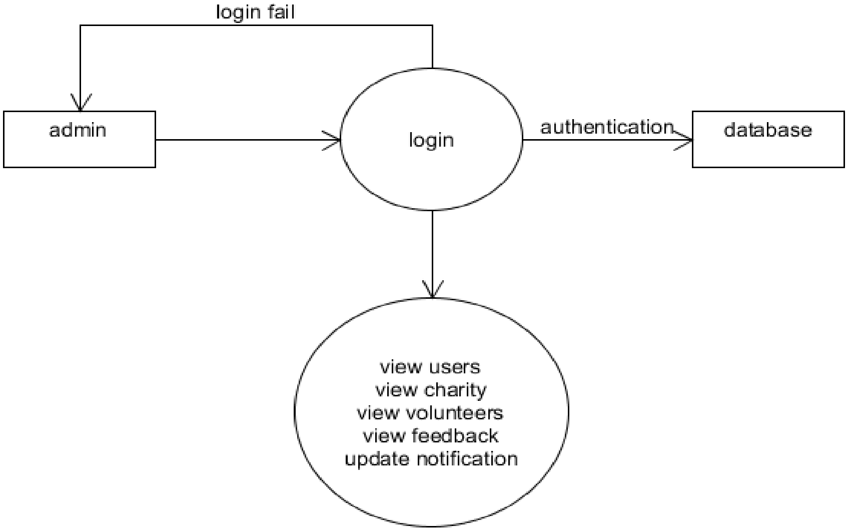
A **data flow diagram** (DFD) is a way of representing a flow of a data of a process or a system (usually an information system) The DFD also provides information about the outputs and inputs of each entity and the process itself. A data flow diagram has no control flow, there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart. There are several notations for displaying data flow diagrams.



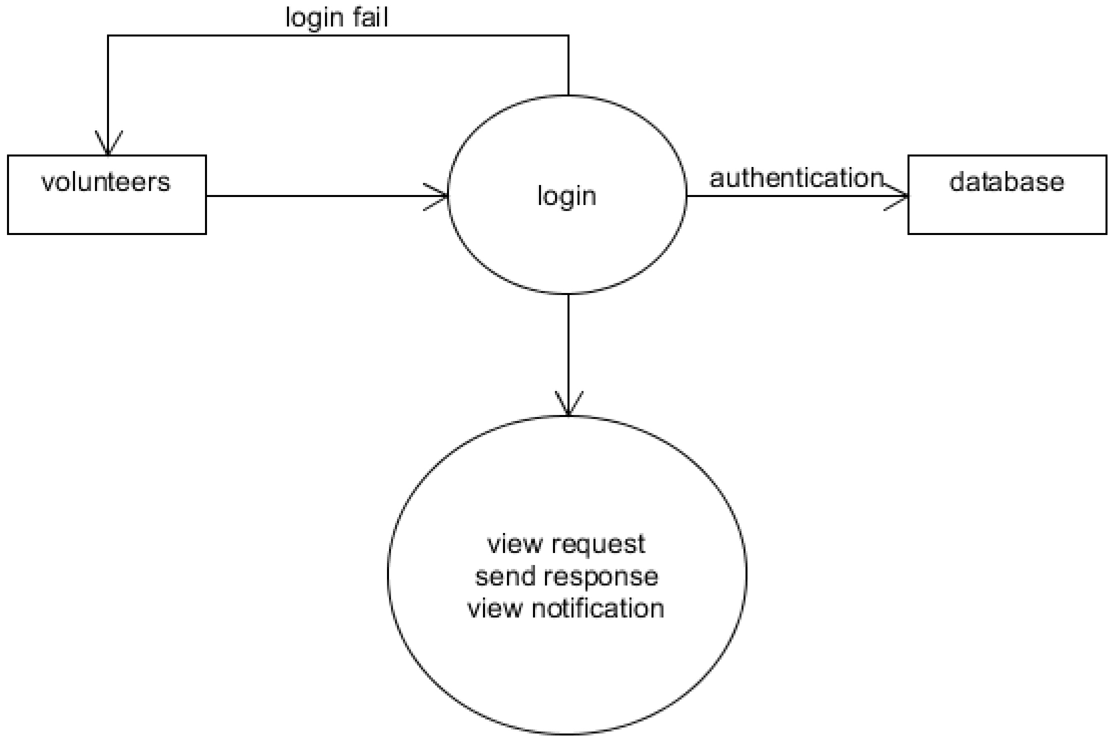
0-Level Data flow Diagram



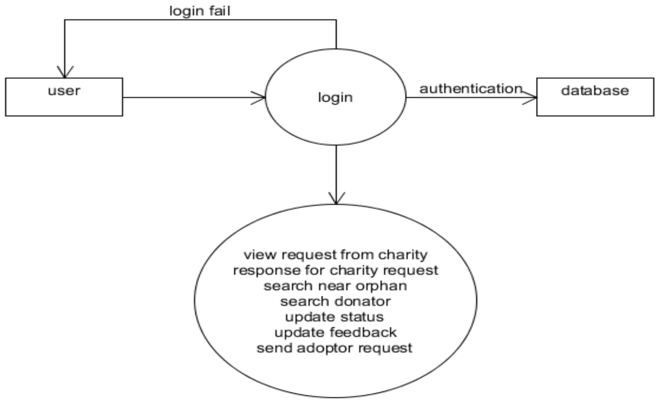
Level 1 : Admin



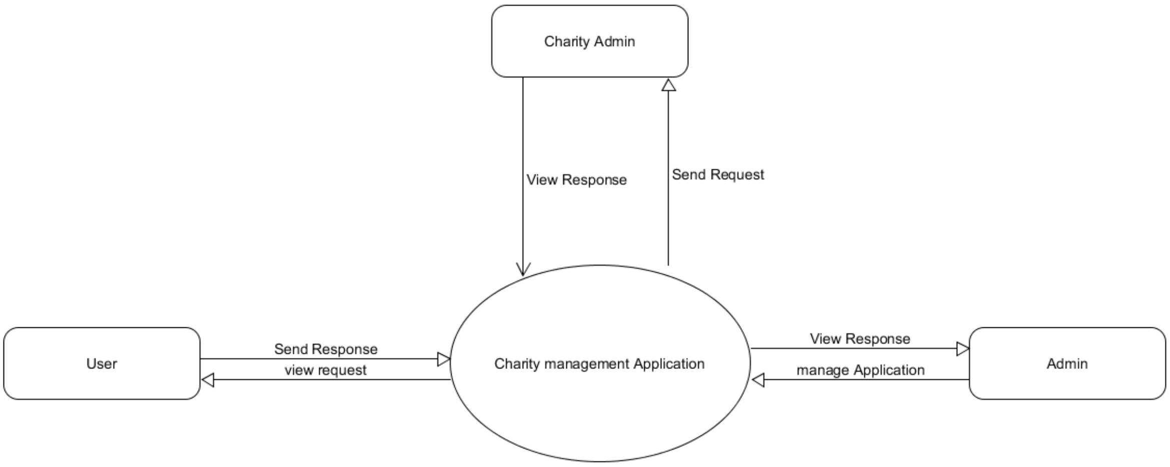
Volunteers



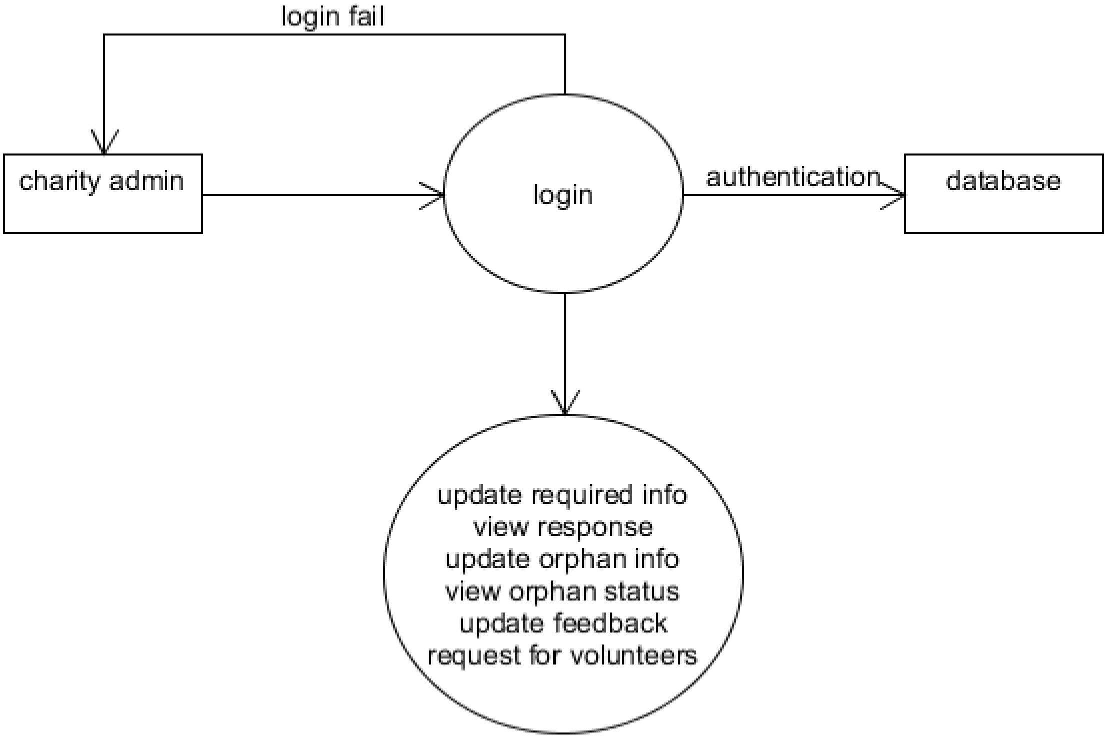
User



Collaboration Diagram



Charity Admin

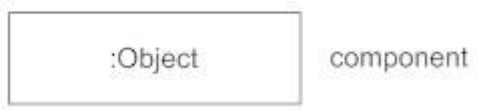


Sequence Diagram

Sequence diagrams describe interactions among classes in terms of an exchange of messages over time. They're also called event diagrams. A sequence diagram is a good way to visualize and validate various runtime scenarios. These can help to predict how a system will behave and to discover responsibilities a class may need to have in the process of modelling a new system.

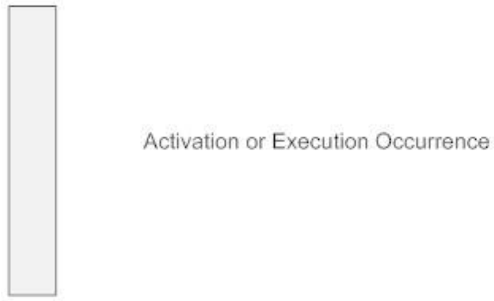
Class Roles or Participants

Class roles describe the way an object will behave in context. Use the UML object symbol to illustrate class roles, but don't list object attributes.



Activation or Execution Occurrence

Activation boxes represent the time an object needs to complete a task. When an object is busy executing a process or waiting for a reply message, use a thin gray rectangle placed vertically on its lifeline.



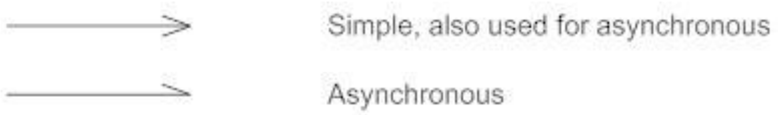
Synchronous Message

A synchronous message requires a response before the interaction can continue. It's usually drawn using a line with a solid arrowhead pointing from one object to another.

Synchronous

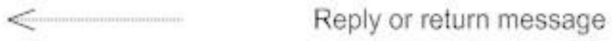
Asynchronous Message

Asynchronous messages don't need a reply for interaction to continue. Like synchronous messages, they are drawn with an arrow connecting two lifelines; however, the arrowhead is usually open and there's no return message depicted.



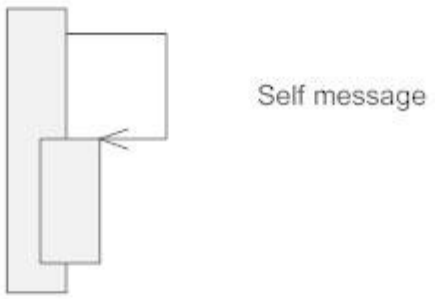
Reply or Return Message

A reply message is drawn with a dotted line and an open arrowhead pointing back to the original lifeline.



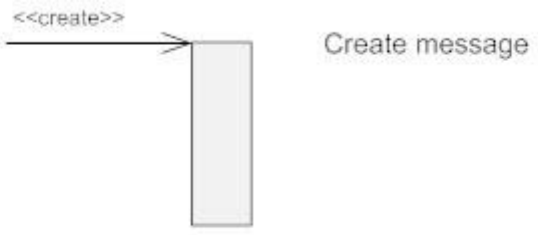
Self Message

A message an object sends to itself, usually shown as a U shaped arrow pointing back to itself.



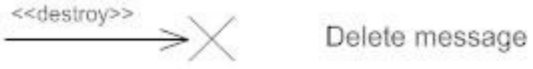
Create Message

This is a message that creates a new object. Similar to a return message, it's depicted with a dashed line and an open arrowhead that points to the rectangle representing the object created.



Delete Message

This is a message that destroys an object. It can be shown by an arrow with an x at the end.



Found Message

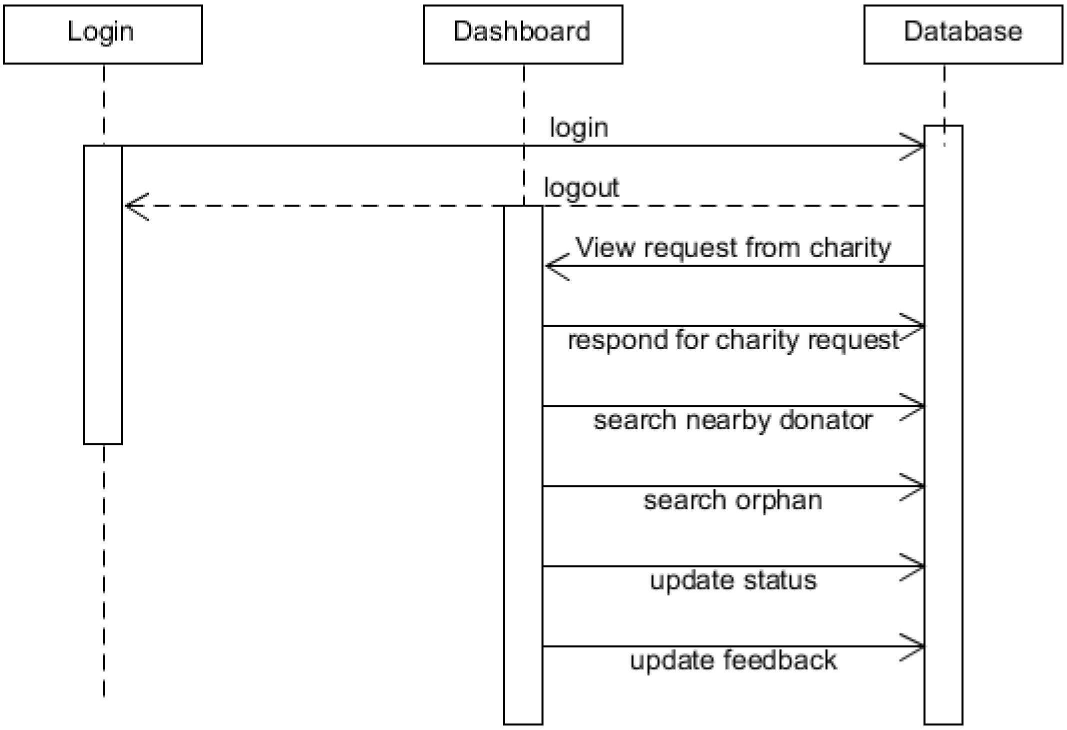
A message sent from an unknown recipient, shown by an arrow from an endpoint to a lifeline.

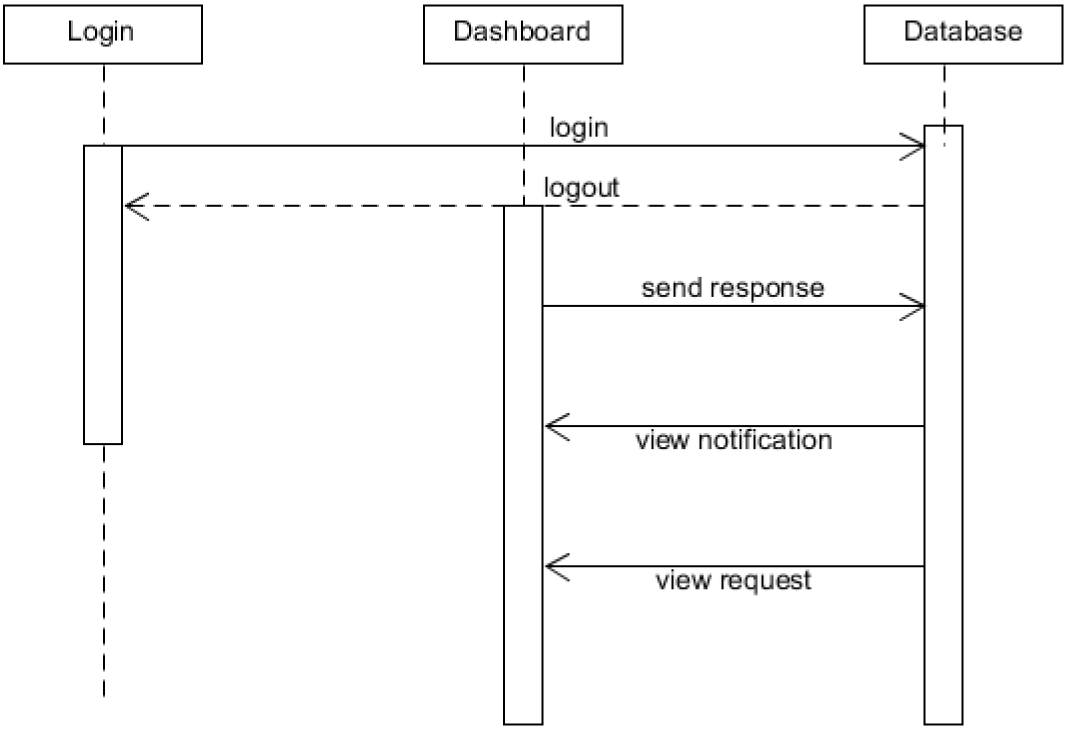
Found message

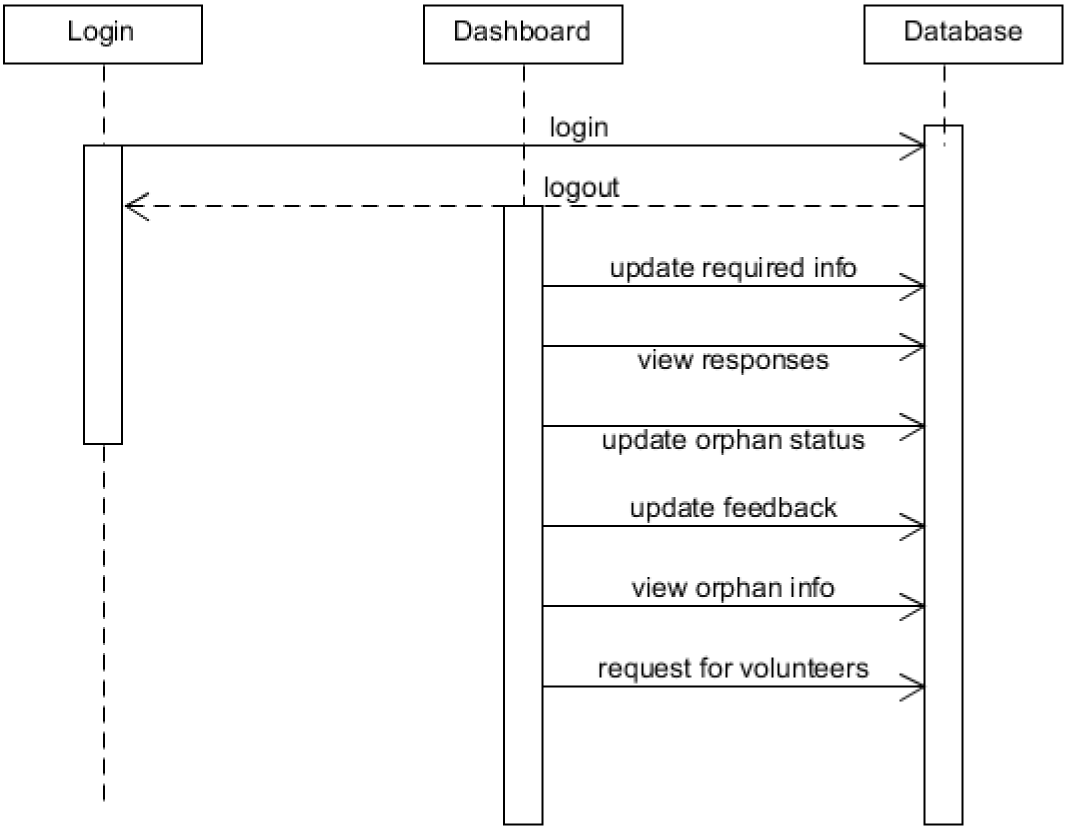
Lost Message

A message sent to an unknown recipient. It's shown by an arrow going from a lifeline to an endpoint, a filled circle or an x.

Lost message




Login
Dashboard
Database
login
logout
send response
view notification
view request


Login
Dashboard
Database
login
logout
update required info
view responses
update orphan status
update feedback
view orphan info
request for volunteers

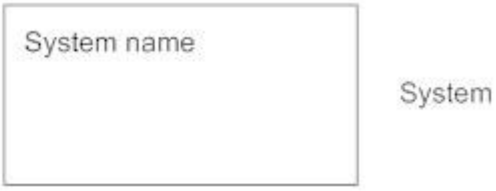
Use case Diagrams:

A **use case diagram** at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses. A use case diagram is a dynamic or behavior diagram in UML. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform.

Notations:

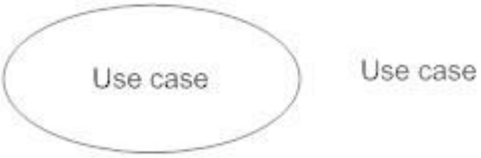
System

Draw your system's boundaries using a rectangle that contains use cases. Place actors outside the system's boundaries.



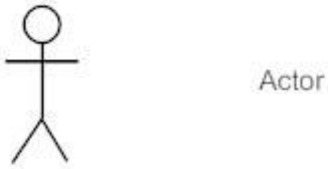
Use Case

Draw use cases using ovals. Label the ovals with verbs that represent the system's functions.



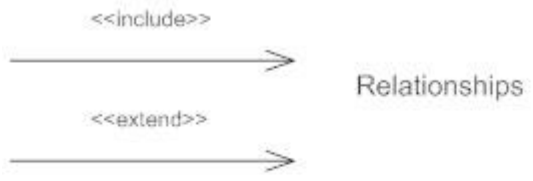
Actors

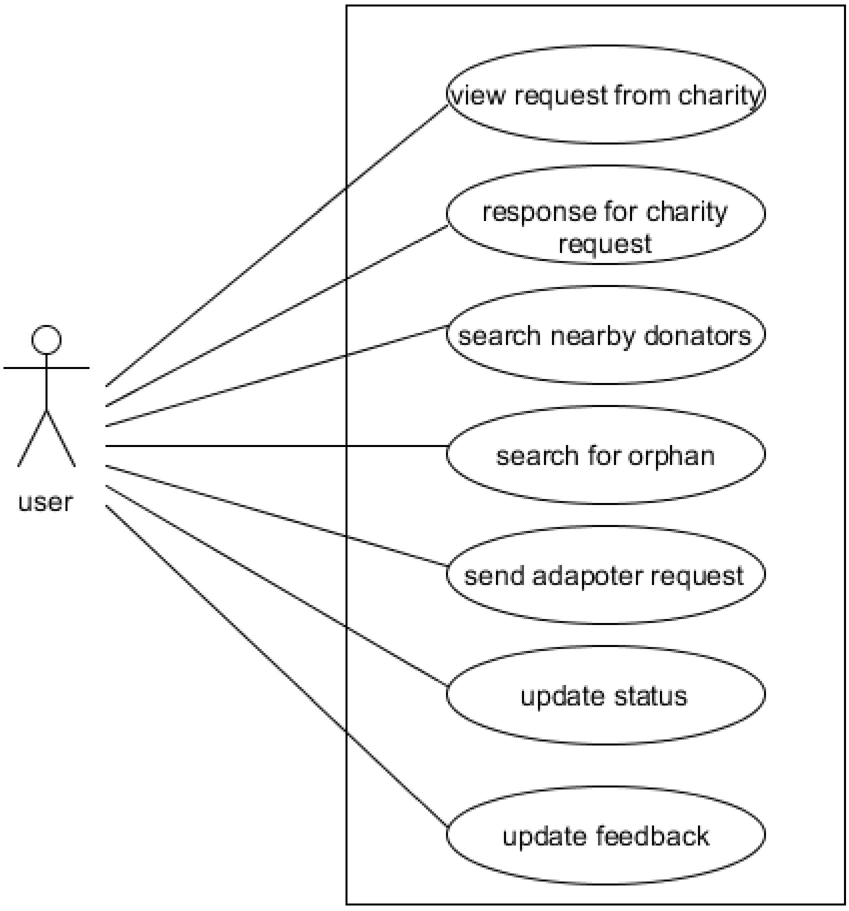
Actors are the users of a system. When one system is the actor of another system, label the actor system with the actor stereotype.

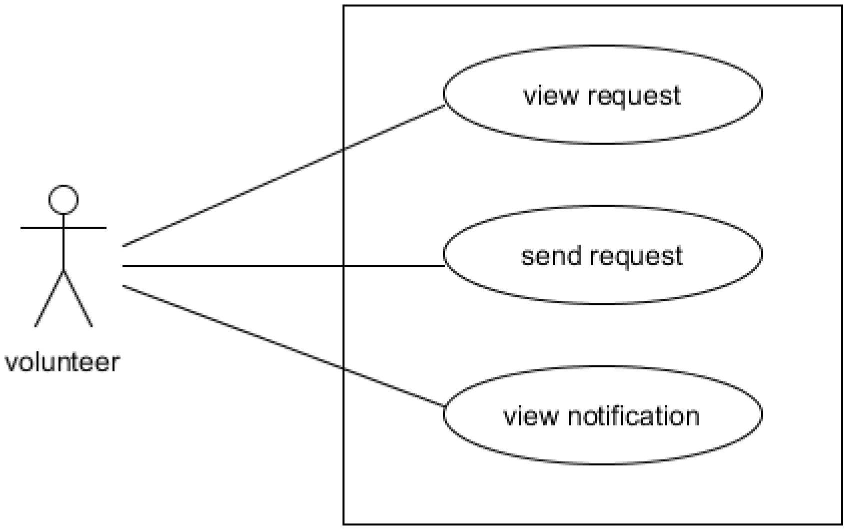


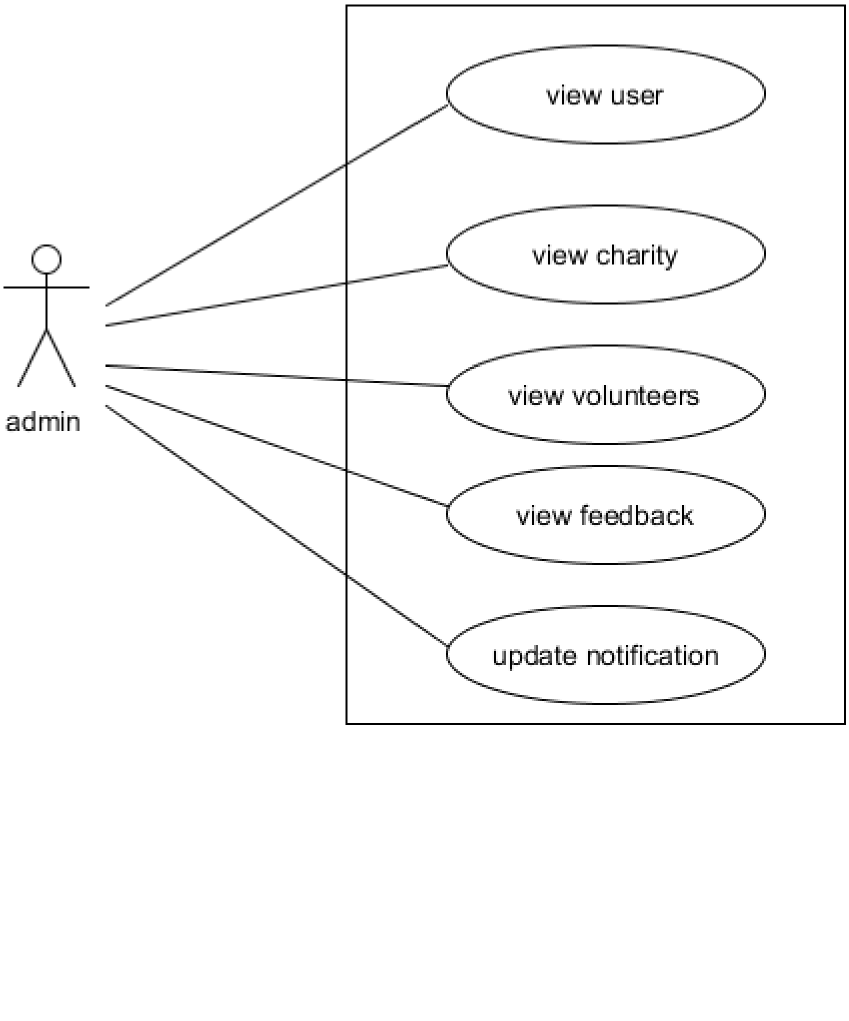
Relationships

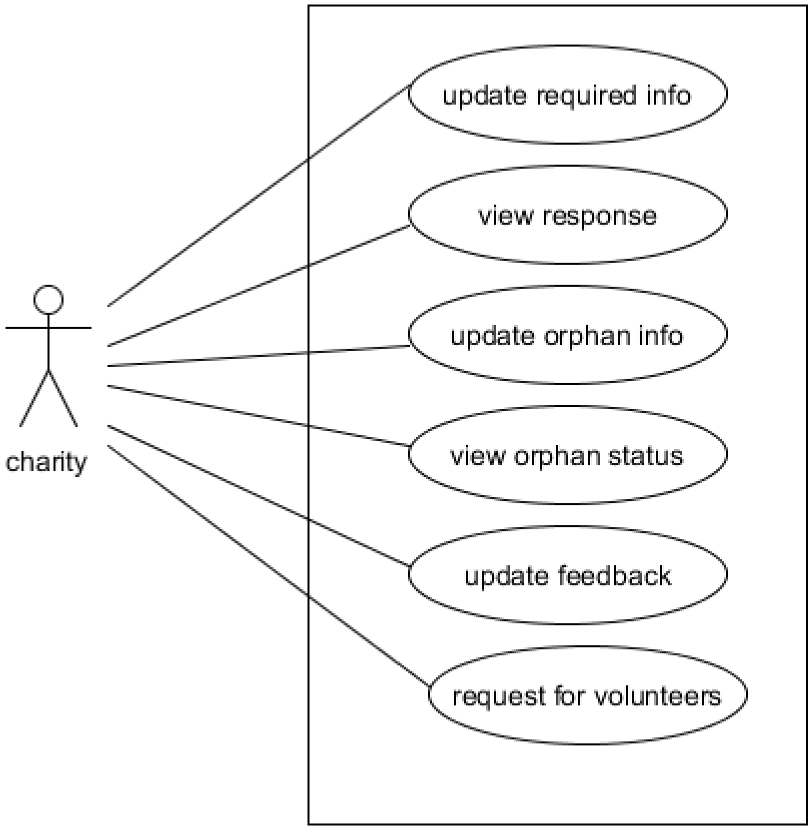
Illustrate relationships between an actor and a use case with a simple line. For relationships among use cases, use arrows labeled either "uses" or "extends." A "uses" relationship indicates that one use case is needed by another in order to perform a task. An "extends" relationship indicates alternative options under a certain use case.





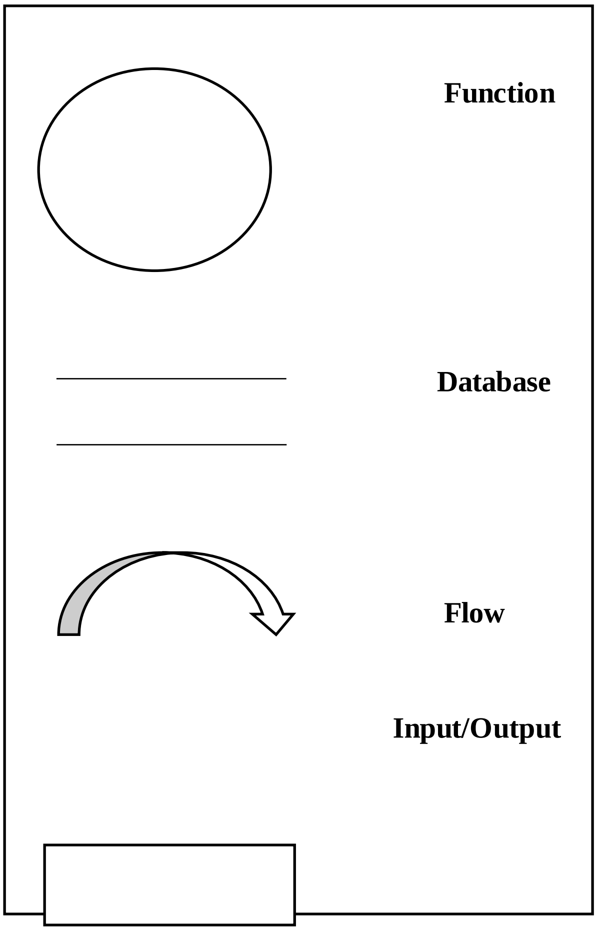

view request
send request
view notification
volunteer

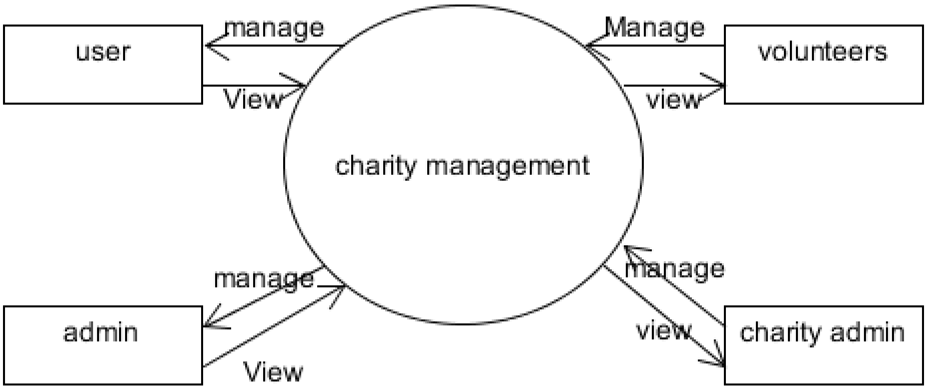

view user
view charity
view volunteers
view feedback
update notification
admin

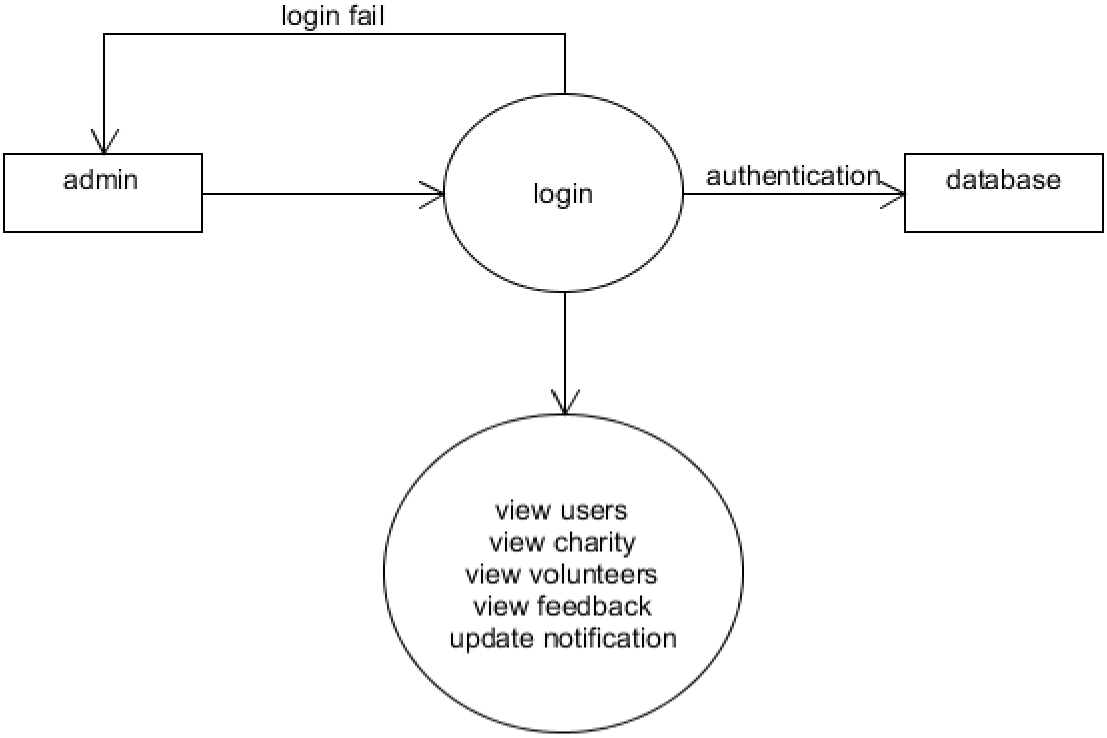

update required info
view response
update orphan info
view orphan status
update feedback
request for volunteers
charity

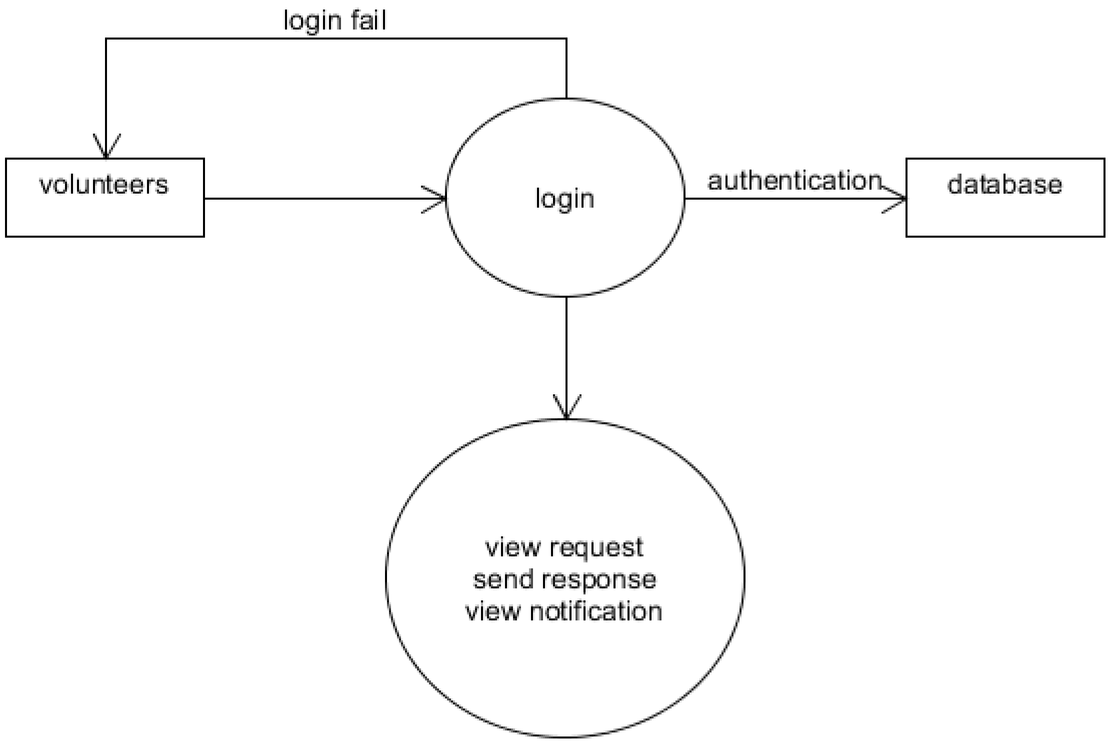
Data Flow Diagram

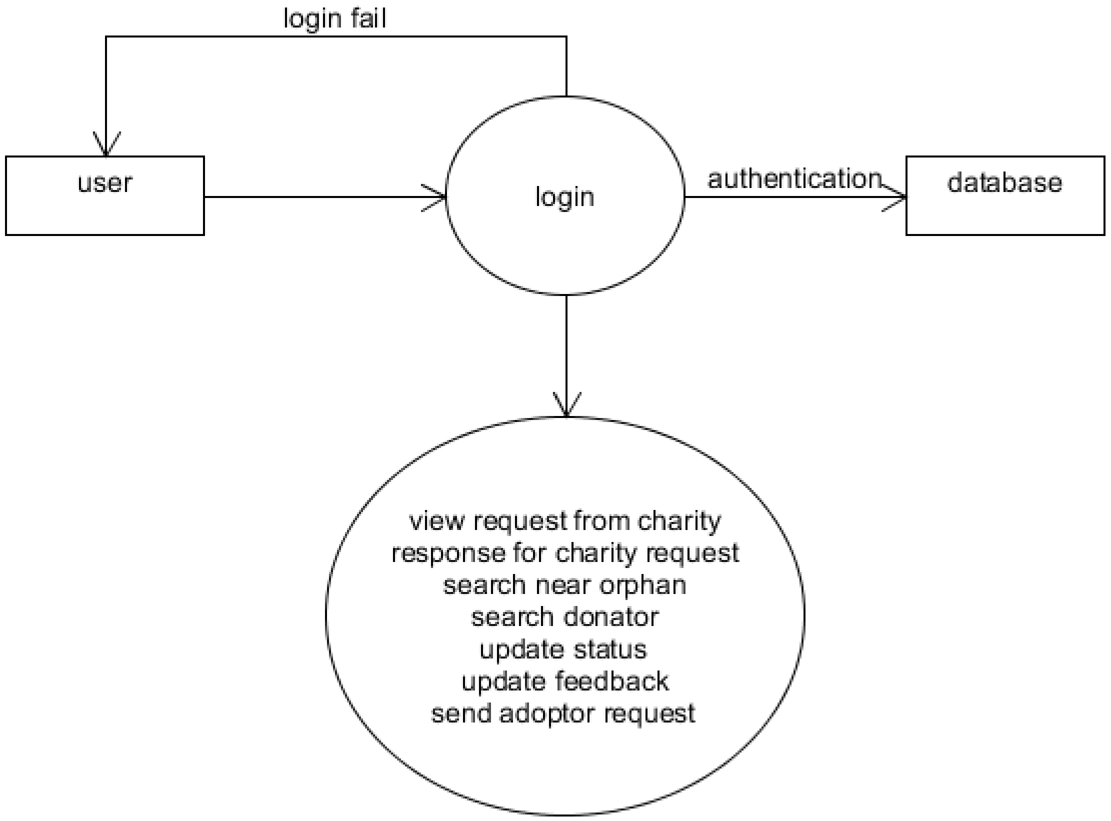
A **data flow diagram** (DFD) is a way of representing a flow of a data of a process or a system (usually an information system) The DFD also provides information about the outputs and inputs of each entity and the process itself. A data flow diagram has no control flow, there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart. There are several notations for displaying data flow diagrams.

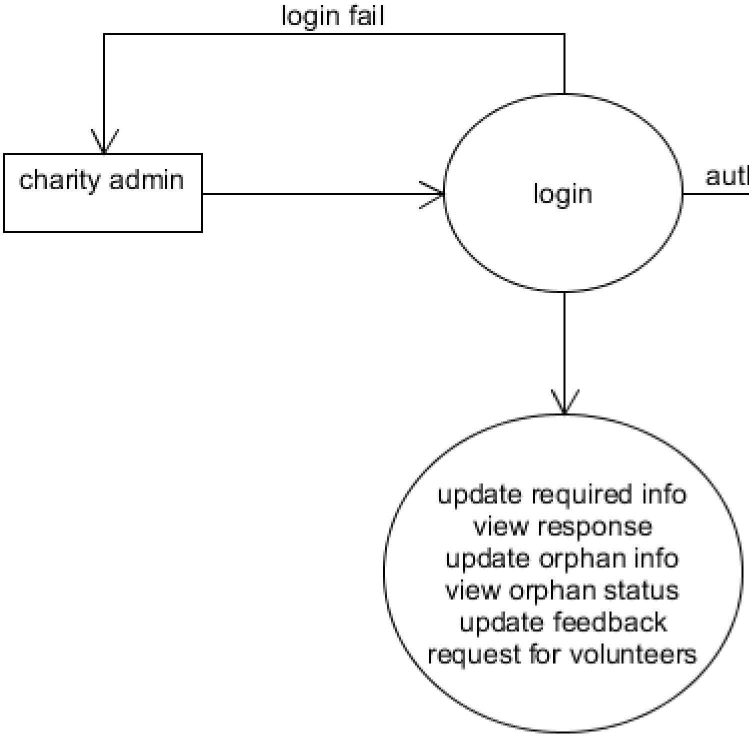



user
manage
View
Manage
volunteers
view
charity management
manage
admin
View
manage
view
charity admin


login fail
admin
login
authentication
database
view users
view charity
view volunteers
view feedback
update notification


login fail
volunteers
login
authentication
database
view request
send response
view notification


login fail
user
login
authentication
database
view request from charity
response for charity request
search near orphan
search donator
update status
update feedback
send adoptor request

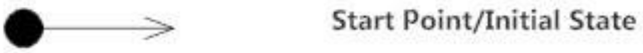

login fail
charity admin
login
authentication
database
update required info
view response
update orphan info
view orphan status
update feedback
request for volunteers

Activity Diagram:

An activity diagram visually presents a series of actions or flow of control in a system similar to a flowchart or a data flow diagram. Activity diagrams are often used in business process modeling. They can also describe the steps in a use case diagram. Activities modeled can be sequential and concurrent. In both cases an activity diagram will have a beginning (an initial state) and an end (a final state).

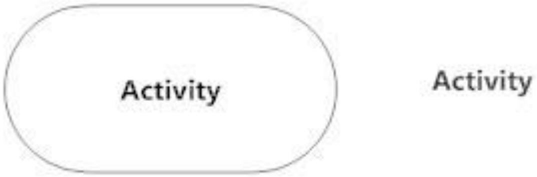
Initial State or Start Point

A small filled circle followed by an arrow represents the initial action state or the start point for any activity diagram. For activity diagram using swimlanes, make sure the start point is placed in the top left corner of the first column.



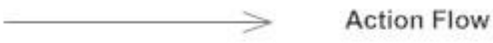
Activity or Action State

An action state represents the non-interruptible action of objects. You can draw an action state in SmartDraw using a rectangle with rounded corners.



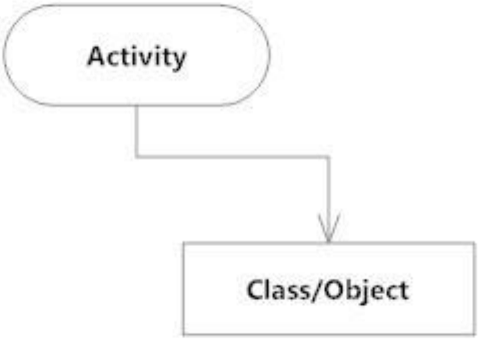
Action Flow

Action flows, also called edges and paths, illustrate the transitions from one action state to another. They are usually drawn with an arrowed line.



Object Flow

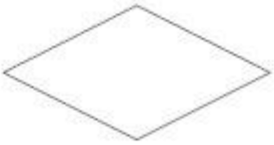
Object flow refers to the creation and modification of objects by activities. An object flow arrow from an action to an object means that the action creates or influences the object. An object flow arrow from an object to an action indicates that the action state uses the object.



Object Flow

Decisions and Branching

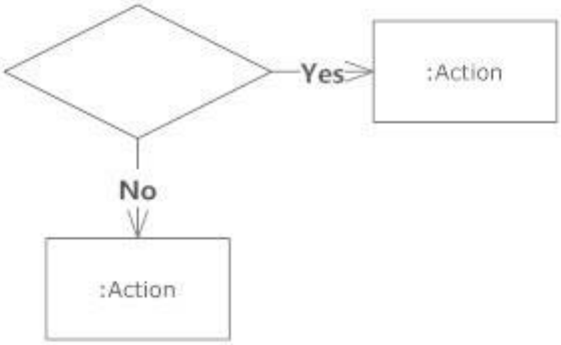
A diamond represents a decision with alternate paths. When an activity requires a decision prior to moving on to the next activity, add a diamond between the two activities. The outgoing alternates should be labeled with a condition or guard expression. You can also label one of the paths "else."



Decision Symbol

Guards

In UML, guards are a statement written next to a decision diamond that must be true before moving next to the next activity. These are not essential, but are useful when a specific answer, such as "Yes, three labels are printed," is needed before moving forward.



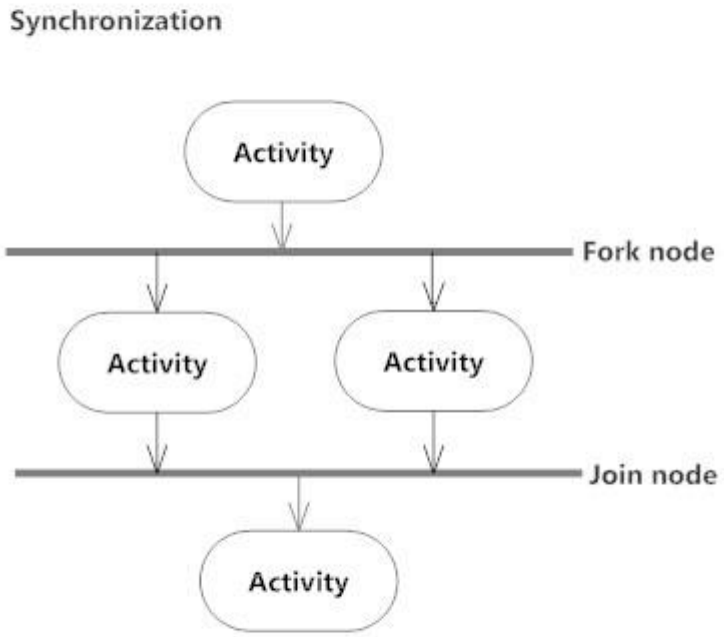
Guard Symbols

Synchronization

A fork node is used to split a single incoming flow into multiple concurrent flows. It is represented as a straight, slightly thicker line in an activity diagram.

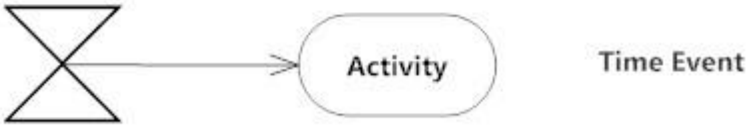
A join node joins multiple concurrent flows back into a single outgoing flow.

A fork and join mode used together are often referred to as synchronization.



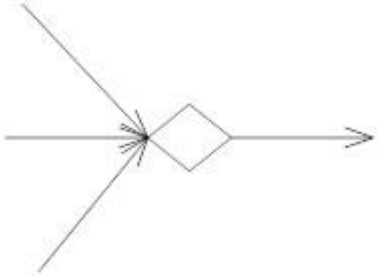
Time Event

This refers to an event that stops the flow for a time; an hourglass depicts it.



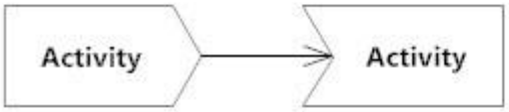
Merge Event

A merge event brings together multiple flows that are not concurrent.



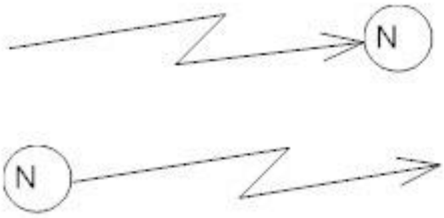
Sent and Received Signals

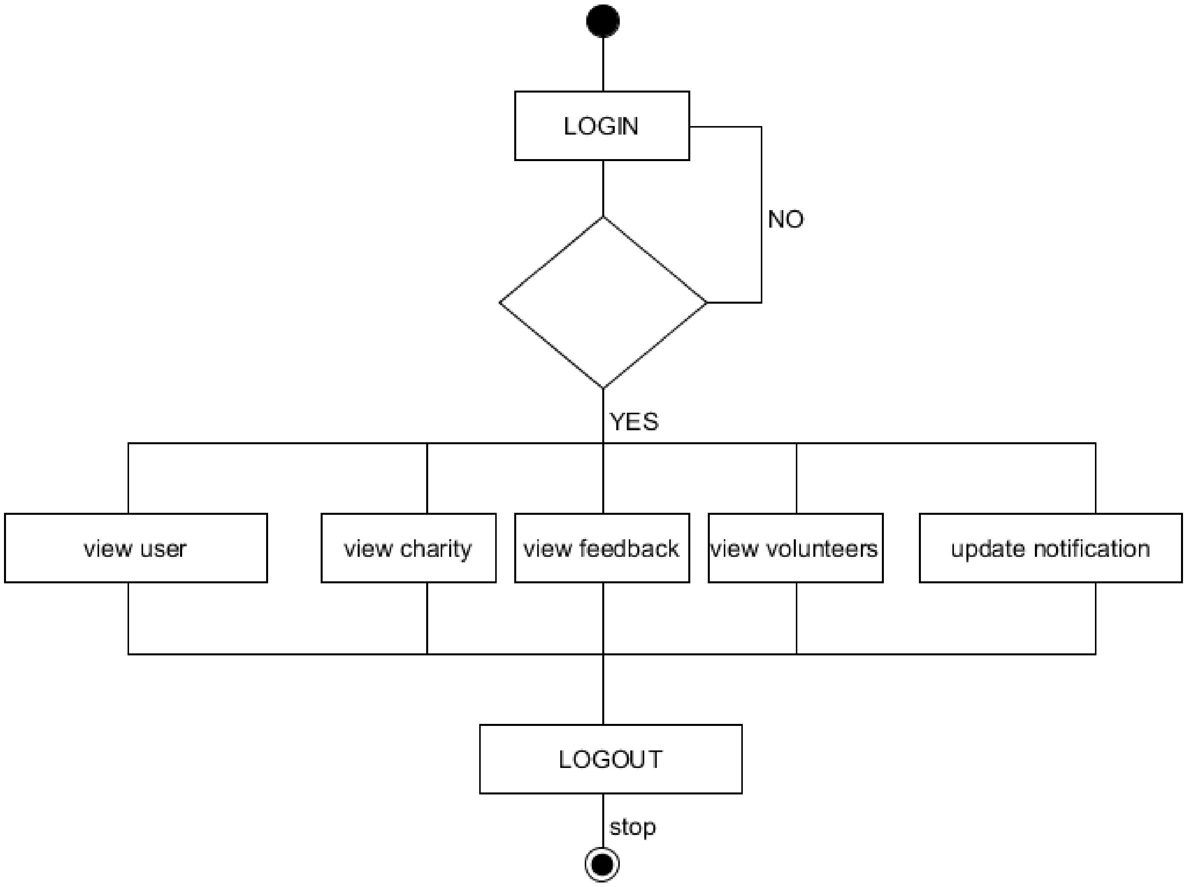
Signals represent how activities can be modified from outside the system. They usually appear in pairs of sent and received signals, because the state can't change until a response is received, much like synchronous messages in a sequence diagram. For example, an authorization of payment is needed before an order can be completed.

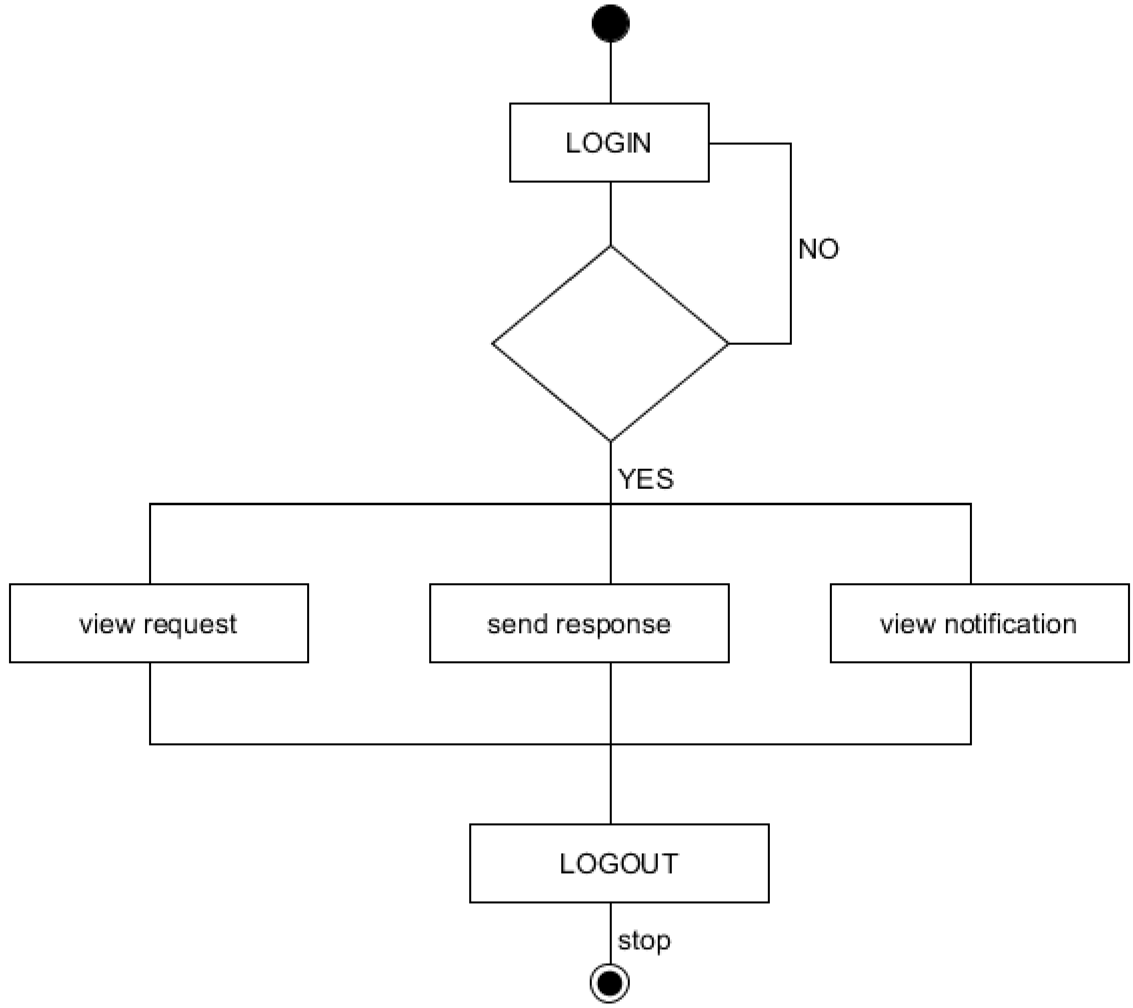


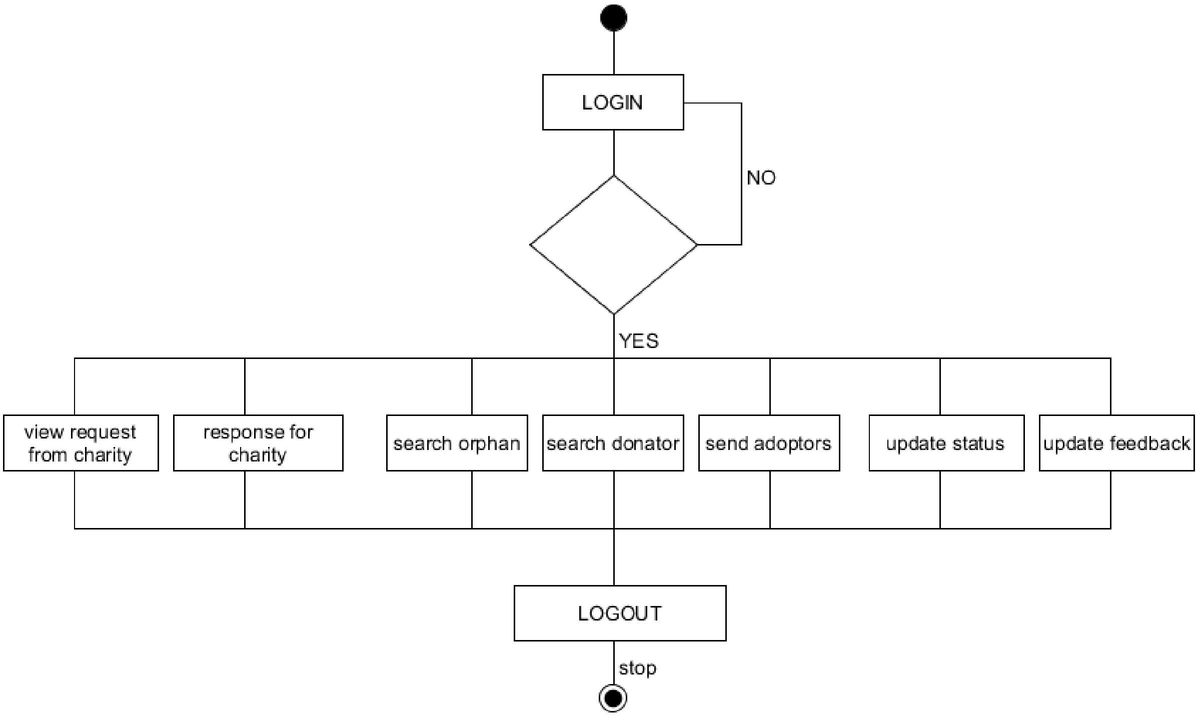
Interrupting Edge

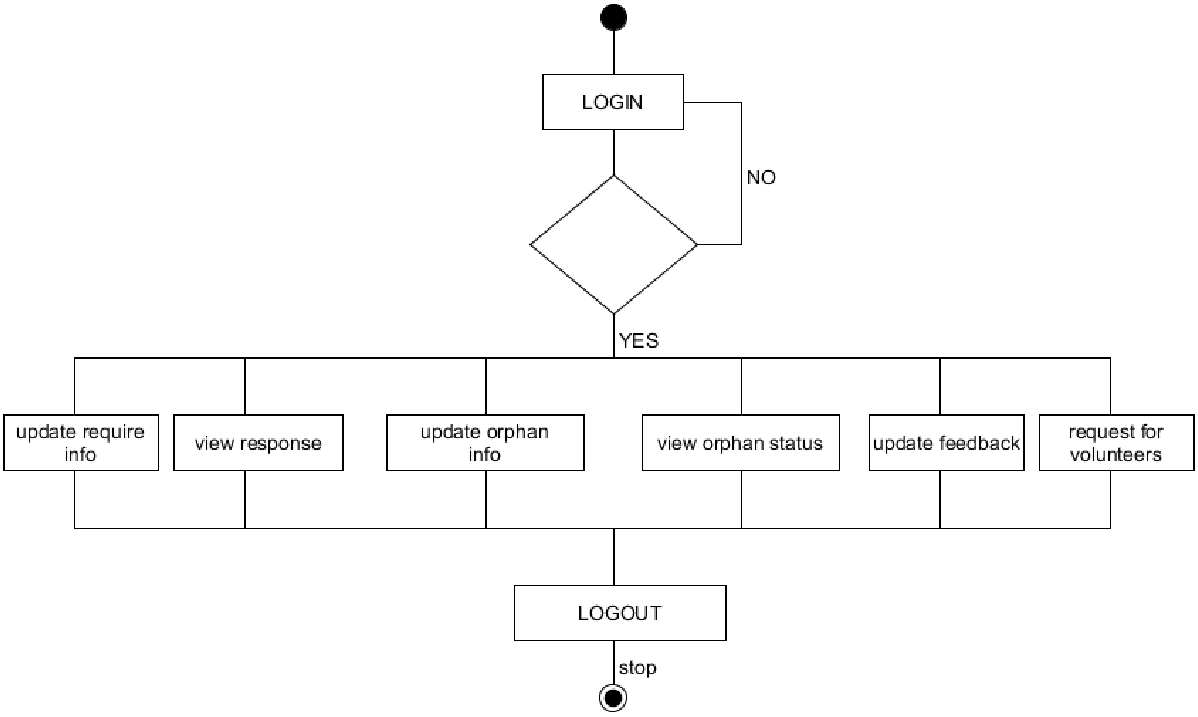
An event, such as a cancellation, that interrupts the flow denoted with a lightning bolt.




LOGIN
NO
YES
view user
view charity
view feedback
view volunteers
update notification
LOGOUT
stop


LOGIN
NO
YES
view request
send response
view notification
LOGOUT
stop

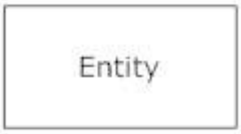

LOGIN
NO
YES
view request from charity
response for charity
search orphan
search donator
send adoptors
update status
update feedback
LOGOUT
stop

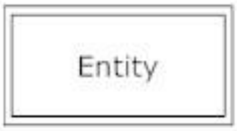

LOGIN
NO
YES
update require info
view response
update orphan info
view orphan status
update feedback
request for volunteers
LOGOUT
stop

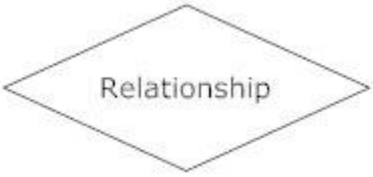
Entity Relationship Diagram

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities. These entities can have attributes that define its properties. By defining the entities, their attributes, and showing the relationships between them, an ER diagram illustrates the logical structure of databases.ER diagrams are used to sketch out the design of a database.

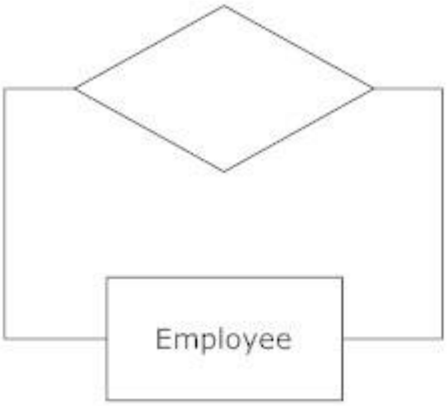
An ER diagram is a means of visualizing how the information a system produces is related. There are five main components of an ERD:

- **Entities**, which are represented by rectangles. An entity is an object or concept about which you want to store information. 

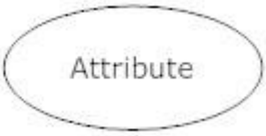
  A weak entity is an entity that must defined by a foreign key relationship with another entity as it cannot be uniquely identified by its own attributes alone. 

- **Actions**, which are represented by diamond shapes, show how two entities share information in the database. 

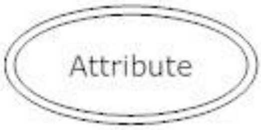
  In some cases, entities can be self-linked. For example, employees can supervise other employees.



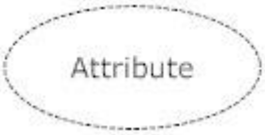
- **Attributes**, which are represented by ovals. A key attribute is the unique, distinguishing characteristic of the entity. For example, an employee's social security number might be the employee's key attribute.

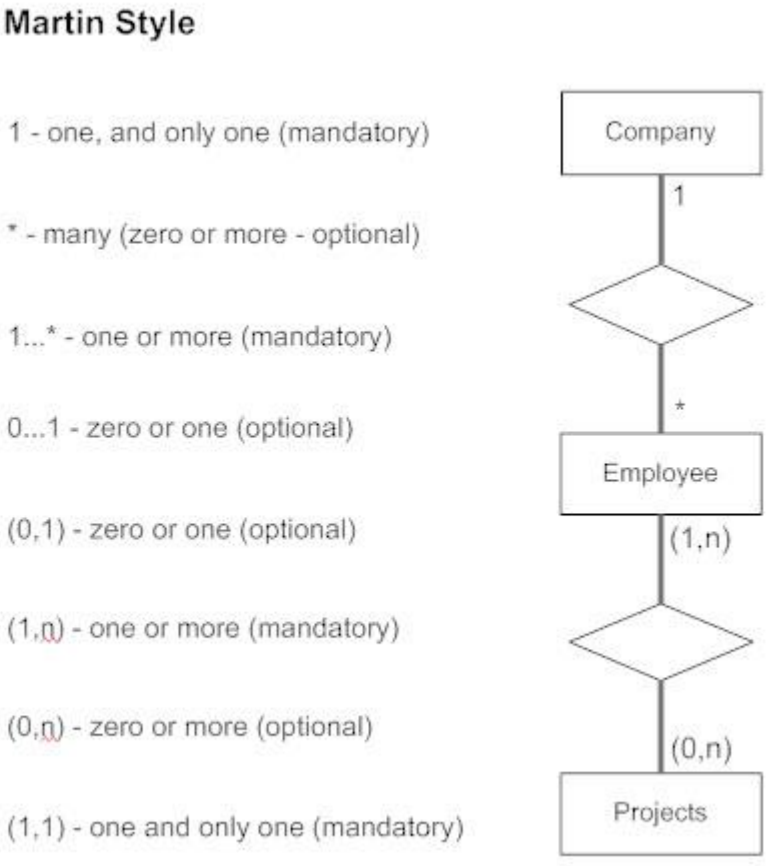


A multivalued attribute can have more than one value. For example, an employee entity can have multiple skill values.



A derived attribute is based on another attribute. For example, an employee's monthly salary is based on the employee's annual salary.





1 - one, and only one (mandatory)

* - many (zero or more - optional)

1...* - one or more (mandatory)

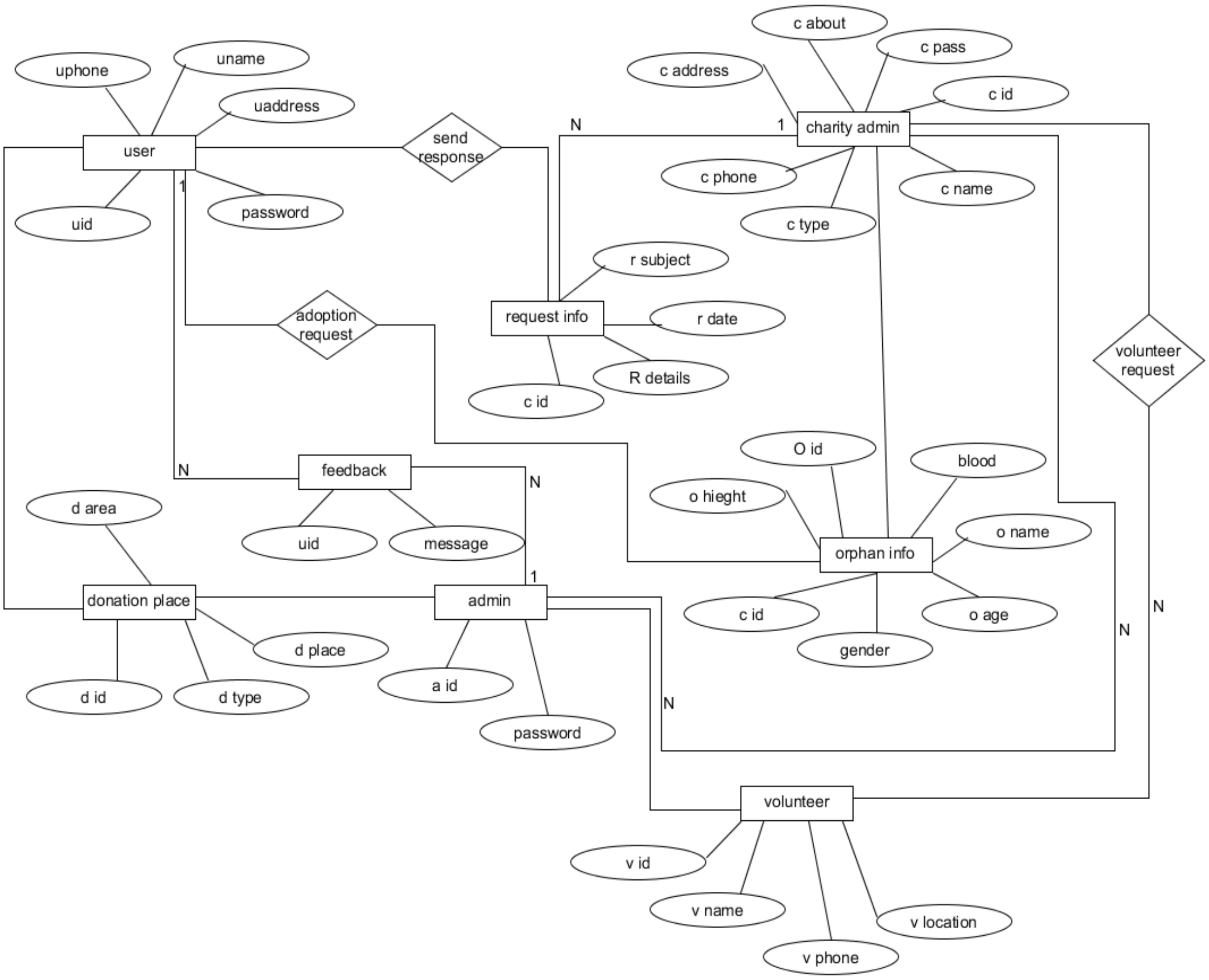
0...1 - zero or one (optional)

(0,1) - zero or one (optional)

(1,n) - one or more (mandatory)

(0,n) - zero or more (optional)

(1,1) - one and only one (mandatory)


c about
c pass
uphone
uname
c address
c id
uaddress
N
1
charity admin
user
send response
c phone
c name
1
uid
password
c type
r subject
adoption request
request info
r date
volunteer request
R details
c id
O id
blood
N
feedback
N
d area
o hieght
o name
uid
message
orphan info
1
donation place
admin
c id
o age
N
d place
N
gender
a id
d id
d type
N
password
volunteer
v id
v name
v location
v phone

Database Schema:

A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data.

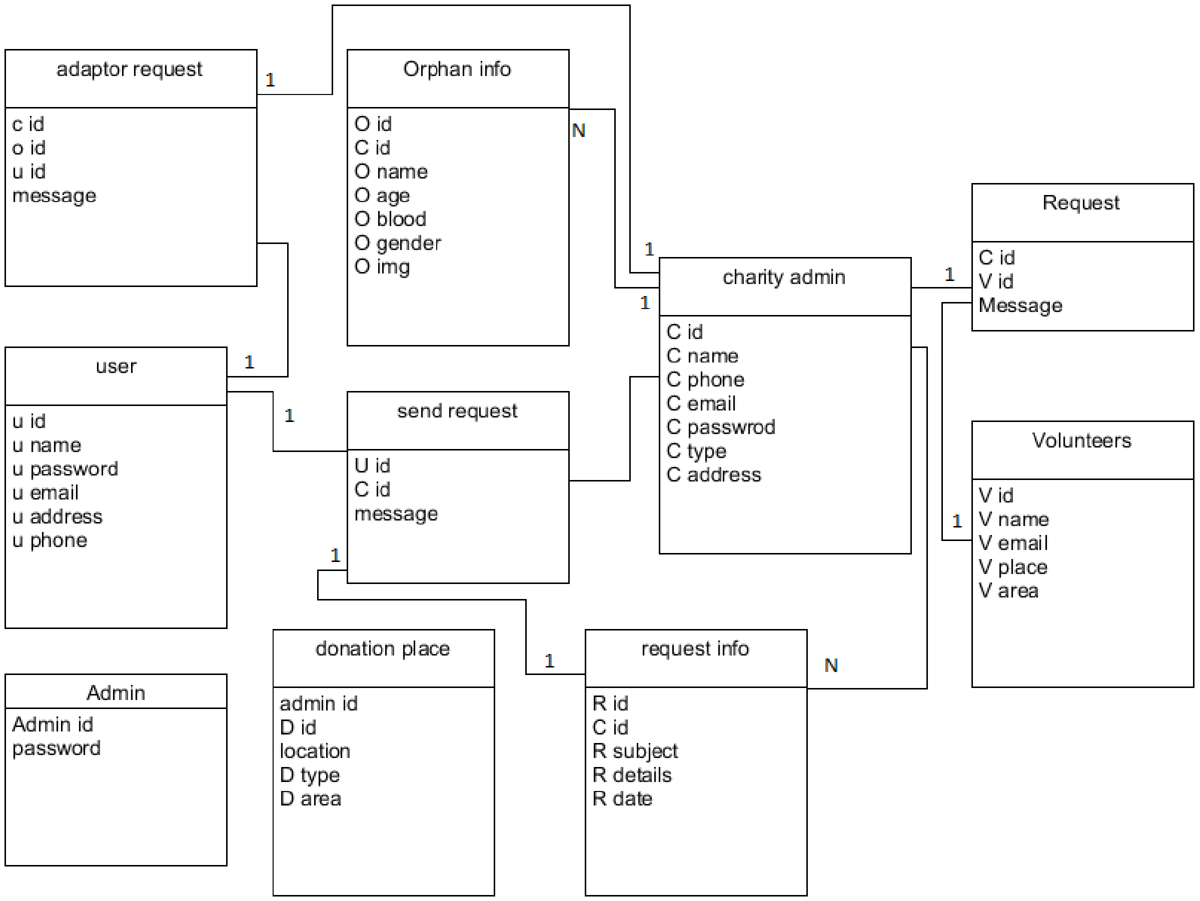
A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams. It's the database designers who design the schema to help programmers understand the database and make it useful.

A database schema can be divided broadly into two categories −

- **Physical Database Schema** − This schema pertains to the actual storage of data and its form of storage like files, indices, etc. It defines how the data will be stored in a secondary storage.
- **Logical Database Schema** − This schema defines all the logical constraints that need to be applied on the data stored. It defines tables, views, and integrity constraints.

Relational Database:

A **relational database** is defined as *a database structured to recognize relations among stored items of information* according to Google search. You can represent data of all sorts through a relational database, such as a grocery store's inventory to a realtor company and their houses.


adaptor request
c id
o id
u id
message
1
Orphan info
O id
C id
O name
O age
O blood
O gender
O img
N
1
1
charity admin
C id
C name
C phone
C email
C passwrod
C type
C address
1
Request
C id
V id
Message
user
u id
u name
u password
u email
u address
u phone
1
1
send request
U id
C id
message
1
Volunteers
V id
V name
V email
V place
V area
1
Admin
Admin id
password
donation place
admin id
D id
location
D type
D area
1
request info
R id
C id
R subject
R details
R date
N

Psudo code

Pseudo code

Registration

```
Form entry ( ); Establish the connection;
Get User Id (email)
Get Password
Get Email
Get Contact Number
If valid, then create a session and prompt, “registered successfully”;
Else
Display appropriate error message;
End.
```

Login Form

```
Data: Username and Password
Result: Create session and Authenticate User
Click on login call to action button on the Orphanage Connecting home page;
Enter registered Username and password;
Connect database
If valid then
Create session on the username;
Authenticate User;
Else

Display error message “Wrong Credentials”;
Redirect to Login Page;
```

Orphanage Profile information

```
Data: Orphanage Profile details
Result: Orphanage Profile should store in database
Fill required details
Enter input field information;
Connect database
```

```
If valid then
Data store in database
Authenticate User input;
Else

Display  message “please enter required field information”;
Redirect to dashboard;
```

System Testing:

INTRODUCTION

Once system implementation is done, system testing will be performed to testing on system performance. System testing is an important process in system development project. It will perform after development process which the actual system or prototype is created. Testing phase is very useful and important because this process able to trickle out the errors inside the software. Normally testing criteria is based on user and system requirements, to verify whether the system meet the requirements or not. System reliability is very important for a system to the end users and testing process able to verify the reliability of the system.

System testing may divide two parts which is unit testing and module integration testing. Unit testing is a kind of testing on each of the individual component in a large system. Before modules integration, unit testing performed on each module able to ensure that every module is working perfectly. The module integration testing would be a testing on the process of combination of all modules. Once all modules able to communicate with each others, the final system is done and the integration testing would test on the complete system.

Types of Test

1.White Box Testing

WHITE BOX TESTING (also known as Clear Box Testing, Open Box Testing, Glass Box Testing, Transparent Box Testing, Code-Based Testing or Structural Testing) is a software testing method in which the internal structure/design/implementation of the item being tested is known to the tester. The tester chooses inputs to exercise paths through the code and determines the appropriate outputs. Programming know-how and the implementation knowledge is essential.

In our charity management system there are many modules which helps the project to build more secure and effective. Before submitting the application to the web browser each modules are tested by the user which are visible for them. If any test case are not worked properly then they are submitted back to developer for the correction.

2.Black Box testing

BLACK BOX TESTING, also known as Behavioural Testing, is a software testing method in which the internal structure/design/implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional.This method is named so because the software program, in the eyes of the tester, is like a black box; inside which one cannot see.

In this charity management application the black box plays the more important role. In black box testing we check for every end user that how the application works well on every browser or not. The developed application should be understood by every end user and for the tester they have to imagine for all such cases and has to do testing for every module.

Levels of Testing:

Acceptance testing:

ACCEPTANCE TESTING is a level of software testing where a system is tested for acceptability. The purpose of this test is to evaluate the system's compliance with the business requirements and assess whether it is acceptable for delivery.

In our charity management testing is used for whether our application run perfectly or not. The application weather meets the user requirement and is it responsive to all the browser and can be acceptable by every end user are checked.

System testing:

SYSTEM TESTING is a level of software testing where a complete and integrated software is tested. The purpose of this test is to evaluate the system's compliance with the specified requirements.

In our application its check for every module because its works well or not according to the user requirements. In our application its checked for test results, and for all the actors are work properly or not according to the assigned work are checked.

Integration Testing:

INTEGRATION TESTING is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing.

In our charity managements project we after doing the unit testing all the individual modules that are pre-processed in the unit testing are gathered or integrated together to check weather this application works perfectly or not. The modules like Admin Module, user module, Volunter module,Donor Module and payment module are merged to see whether they work or not.

Unit Testing:

UNIT TESTING is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output. In procedural programming, a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or derived/ child class. (Some treat a module of an application as a unit. This is to be discouraged as there will probably be many individual units within that module.)

In our application there are many modules like Admin Module, user module, Volunter module,Donor Module each module are tested based on the requirements gathered for this application which helps the application works perfectly. The every actors modules are analysed and tested for each operation and the results are shown.

Positive Test Case

Test Case No	Positive Scenario	Required Input	Expected Result	Actual Output	Test pass/fail
1	To ensure that the module able to update registration Details	Admin, Donor User and Volunteer	The registration status should done accordingly	As expected	Pass
2	To ensure the module able to accept login	Username or password	After successful it navigate to home page	As expected	Pass
3	To ensure the module able to display all status admin	Donors details, user details, and Volunteer details	The status should update accordingly.	As expected	Pass
4	Feedback to Admin	Feedback	Status updated accordingly	As expected	Pass
5	To ensure that module able to send request to the Donor	Send Request to Donor	Details are showed to the Donor	As expected	Pass
6	To ensure that module able to Accept request to the Donor	Recieve Request to Donor	Details are showed to the Admin	As expected	Pass

7	To ensure the module able to update response from user	Response status are updated by Temple	All status should update accordingly.	As expected	Pass
8	To ensure the module able provide recipt to the donor	Donators reciept	The status should update accordingly.	As expected	Pass
9	To ensure the module able to provide campaign details	Make campaign	The status should be uploaded	As expected	Pass
10	To ensure the module able to update the user requirement	Make request	The status should be uploaded	As expected	Pass

Negative Test Cases

Test Case No	**Negative Scenario**	**Required Input**	**Expected Result**	**Actual Output**	**Test pass/fail**
1	Trying to perform login operations	Username and password	The application is render to main page	Due to database issues there will be no navigation	fail

2	Trying To ensure the module able to display all register details	Details of user and Admin, Donor, user	The status should be displayed on screen	Due to database issues there will be no display of data	fail
3	Trying To ensure that the module able to update charity details	Charity information	The charity and status should update accordingly.	Due to database error there will no data display	fail

Conclusion:

The proposed system presents ,a new internet-based application that provides a platform for donating leftover food to all needy people/organisations. The system is shown to be an effective means of donating things to organisations, etc. over the internet. It shows the potential for avoiding the wastage of food. It provides information about how the product works for betterment of the society. This system will create a common collaboration portal for hotels/restaurants and charities, charity can directly contact restaurants who have food remaining. This product is a web application which aims to establish a link between restaurants and the charity homes/needy households to enable excess food donation. In the current scenario food is being wasted daily on a large basis in different restaurants, weddings, social functions, college canteens and many other social events. This projects also helps in helding campaigns like blood donation or creating awareness among the food. People donate food manually by visiting each organisation number of times in order to reduce the problem of food wastage where as there are websites that have taken efforts to help people donate food

Future Enhancement:

The charity management it is web based application we can build it for the mobile application so that its helps in the easy communication to the users. the charity management application can be done more effectively by adding extra modules like the requesting for the big companies for the found or celebrities etc. the following are the features we can add into the application.

- We can implement crossplatform application in both android and Iphone Application
- We can also enable payment gateway like Google pay, Phone pay ETC
- Adding more feature like Location traking and Donation Pickup service also helps to manage appliction effectively

References

1. *Kamal Acharya. School management system project report. Authorea. August 01, 2024. DOI: https://doi.org/10.22541/au.172254873.34023165/v1*

2. *Kamal Acharya. A CASE STUDY OF CINEMA MANAGEMENT SYSTEM PROJECT. Authorea. August 01, 2024. DOI: https://doi.org/10.22541/au.172254873.30191075/v1*

3. *Kamal Acharya. A CASE STUDY ON ONLINE TICKET BOOKING SYSTEM PROJECT. Authorea. August 01, 2024. DOI: https://doi.org/10.22541/au.172254872.26972790/v1*

4. *Kamal Acharya. Web chatting application project report management system. Authorea. August 01, 2024. DOI: https://doi.org/10.22541/au.172254871.18588592/v1*

5. *Kamal Acharya. RETAIL STORE MANAGEMENT SYSTEM PROJECT REPORT. Authorea. August 01, 2024. DOI: https://doi.org/10.22541/au.172254871.14590154/v1*

6. *Kamal Acharya. SUPERMARKET MANAGEMENT SYSTEM PROJECT REPORT. Authorea. August 01, 2024. DOI: https://doi.org/10.22541/au.172252491.19145062/v1*

7. *Kamal Acharya. SOCIAL MEDIA MANAGEMENT SYSTEM PROJECT REPORT. Authorea. August 01, 2024. DOI: https://doi.org/10.22541/au.172252491.11210579/v1*

8. *Kamal Acharya. Online music portal management system project report. Authorea. August 01, 2024. DOI: https://doi.org/10.22541/au.172252488.89734698/v1*

9. *Kamal Acharya. COLLEGE BUS MANAGEMENT SYSTEM PROJECT REPORT. Authorea. July 31, 2024. DOI: https://doi.org/10.22541/au.172245277.70798942/v1*

10. *Kamal Acharya. AUTOMOBILE MANAGEMENT SYSTEM PROJECT REPORT. Authorea. July 31, 2024. DOI: https://doi.org/10.22541/au.172245276.67982593/v1*

11. *Kamal Acharya. Ludo management system project report. Authorea. July 31, 2024. DOI: https://doi.org/10.22541/au.172243999.98091616/v1*

12. *Kamal Acharya. Literature online quiz system project report. Authorea. July 31, 2024 DOI: https://doi.org/10.22541/au.172243825.53562953/v1*

13. *Kamal Acharya. Avoid waste management system project. Authorea. July 29, 2024. DOI: https://doi.org/10.22541/au.172228528.85022205/v1*

14. *Kamal Acharya. CHAT APPLICATION THROUGH CLIENT SERVER MANAGEMENT SYSTEM PROJECT. Authorea. July 29, 2024. DOI: https://doi.org/10.22541/au.172228527.74316529/v1*

15. *Acharya, Kamal, Online Job Portal Management System (May 5, 2024). Available at SSRN: https://ssrn.com/abstract=4817534* or *http://dx.doi.org/10.2139/ssrn.4817534*

16. *Acharya, Kamal, Employee leave management system. (May 7, 2024). Available at SSRN: https://ssrn.com/abstract=4819626* or *http://dx.doi.org/10.2139/ssrn.4819626*

17. *Acharya, Kamal, Online electricity billing project report. (May 7, 2024). Available at SSRN: https://ssrn.com/abstract=4819630* or *http://dx.doi.org/10.2139/ssrn.4819630*

18. *Acharya, Kamal, POLICY MANAGEMENT SYSTEM PROJECT REPORT. (December 10, 2023). Available at SSRN: https://ssrn.com/abstract=4831694* or *http://dx.doi.org/10.2139/ssrn.4831694*

19. *Acharya, Kamal, Software testing for project report. (May 16, 2023). Available at SSRN: https://ssrn.com/abstract=4831028* or *http://dx.doi.org/10.2139/ssrn.4831028*

20. *Acharya, Kamal, ONLINE CRIME REPORTING SYSTEM PROJECT. (August 10, 2022). Available at SSRN: https://ssrn.com/abstract=4831015* or *http://dx.doi.org/10.2139/ssrn.4831015*

21. *Acharya, Kamal, Burger ordering system project report. (October 10, 2022). Available at SSRN: https://ssrn.com/abstract=4832704* or *http://dx.doi.org/10.2139/ssrn.4832704*

23. *Acharya, Kamal, Teachers Record Management System Project Report (December 10, 2023). Available at SSRN: https://ssrn.com/abstract=4833821* or *http://dx.doi.org/10.2139/ssrn.4833821*

23. *Acharya, Kamal, Dairy Management System Project Report (December 20, 2020). Available at SSRN: https://ssrn.com/abstract=4835231* or *http://dx.doi.org/10.2139/ssrn.4835231*

24. *Acharya, Kamal, Electrical Shop Management System Project (December 10, 2019). Available at SSRN: https://ssrn.com/abstract=4835238* or *http://dx.doi.org/10.2139/ssrn.4835238*

25. *Acharya, Kamal, Online book store management system project report. (Febuary 10, 2020). Available at SSRN: https://ssrn.com/abstract=4835277* or *http://dx.doi.org/10.2139/ssrn.4835277*

26. *Acharya, Kamal, Paint shop management system project report. (January 10, 2019). Available at SSRN: https://ssrn.com/abstract=4835441* or *http://dx.doi.org/10.2139/ssrn.4835441*

27. *Acharya, Kamal, Supermarket billing system project report. (August 10, 2021). Available at SSRN: https://ssrn.com/abstract=4835474* or *http://dx.doi.org/10.2139/ssrn.4835474*

28. *Acharya, Kamal, Online taxi booking system project report. (March 10, 2022). Available at SSRN: https://ssrn.com/abstract=4837729* or *http://dx.doi.org/10.2139/ssrn.4837729*

29. *Acharya, Kamal, Online car servicing system project report. (March 10, 2023). Available at SSRN: https://ssrn.com/abstract=4837832* or *http://dx.doi.org/10.2139/ssrn.4837832*

30. *Acharya, Kamal, School management system project report. (July 10, 2021). Available at SSRN: https://ssrn.com/abstract=4837837* or *http://dx.doi.org/10.2139/ssrn.4837837*

31. *Acharya, Kamal, Furniture Showroom Management System Project Report (March 21, 2021). Available at SSRN: https://ssrn.com/abstract=4839422* or *http://dx.doi.org/10.2139/ssrn.4839422*

32. *Acharya, Kamal, Online Vehicle Rental System Project Report (March 21, 2019). Available at SSRN: https://ssrn.com/abstract=4839429* or *http://dx.doi.org/10.2139/ssrn.4839429*

33. *Acharya, Kamal, Fruit Shop Management System Project Report (August 10, 2023). Available at SSRN: https://ssrn.com/abstract=4841048* or *http://dx.doi.org/10.2139/ssrn.4841048*

34. *Acharya, Kamal, Hall Booking Management System Project Report (December 21, 2023). Available at SSRN: https://ssrn.com/abstract=4841055* or *http://dx.doi.org/10.2139/ssrn.4841055*

35. *Acharya, Kamal, Lundry Management System Project Report (October 21, 2023). Available at SSRN: https://ssrn.com/abstract=4841059* or *http://dx.doi.org/10.2139/ssrn.4841059*

36. *Acharya, Kamal, A CASE STUDY OF CINEMA MANAGEMENT SYSTEM PROJECT (September 25, 2023). Available at SSRN: https://ssrn.com/abstract=4841209* or *http://dx.doi.org/10.2139/ssrn.4841209*

37. *Acharya, Kamal, A CASE STUDY ON ONLINE TICKET BOOKING SYSTEM PROJECT (May 25, 2024). Available at SSRN: https://ssrn.com/abstract=4841210* or *http://dx.doi.org/10.2139/ssrn.4841210*

38. *Acharya, Kamal, ONLINE DATING MANAGEMENT SYSTEM PROJECT REPORT. (April 25, 2023). Available at SSRN: https://ssrn.com/abstract=4842066 or http://dx.doi.org/10.2139/ssrn.4842066*

39. *Acharya, Kamal, ONLINE RESUME BUILDER MANAGEMENT SYSTEM PROJECT REPORT. (April 25, 2021). Available at SSRN: https://ssrn.com/abstract=4842071 or http://dx.doi.org/10.2139/ssrn.4842071*

40. *Acharya, Kamal, TOLL TEX MANAGEMENT SYSTEM PROJECT REPORT (August 21, 2023). Available at SSRN: https://ssrn.com/abstract=4842082 or http://dx.doi.org/10.2139/ssrn.4842082*

41. *Acharya, Kamal, Chat Application Through Client Server Management System Project Report (June 25, 2023). Available at SSRN: https://ssrn.com/abstract=4842761 or http://dx.doi.org/10.2139/ssrn.4842761*

42. *Acharya, Kamal, Web Chatting Application Management System Project Report (April 25, 2022). Available at SSRN: https://ssrn.com/abstract=4842771 or http://dx.doi.org/10.2139/ssrn.4842771*

43. *Acharya, Kamal, Automobile management system project report (May 25, 2022). Available at SSRN: https://ssrn.com/abstract=4846917 or http://dx.doi.org/10.2139/ssrn.4846917*

44. *Acharya, Kamal, College bus management system project report (April 25, 2023). Available at SSRN: https://ssrn.com/abstract=4846920 or http://dx.doi.org/10.2139/ssrn.4846920*

45. *Acharya, Kamal, Courier management system project report (May 25, 2023). Available at SSRN: https://ssrn.com/abstract=4846922 or http://dx.doi.org/10.2139/ssrn.4846922*

46. *Acharya, Kamal, Event management system project report (April 25, 2021). Available at SSRN: https://ssrn.com/abstract=4846927 or http://dx.doi.org/10.2139/ssrn.4846927*

47. *Acharya, Kamal, Library management system project report II (May 25, 2020). Available at SSRN: https://ssrn.com/abstract=4848857 or http://dx.doi.org/10.2139/ssrn.4848857*

48. *Kamal Acharya. Online directory management system project. Authorea. Jun 11, 2025 DOI: https://doi.org/10.22541/au.174965566.61058065/v1*